SIGIR 2018 Tutorial
July 8, 2018
Ann Arbor, USA

Deep Learning for Matching in Search and Recommendation

Jun Xu
Chinese Academy of Sciences

Xiangnan He
National University of Singapore

Hang Li
Bytedance AI Lab

Slides: http://comp.nus.edu.sg/~xiangnan/sigir18-deep.pdf

Outline of Tutorial

- Unified View of Matching in Search and Recommendation
- Part 1: Traditional Approaches to Matching
- Part 2: Deep Learning Approaches to Matching
- Summary

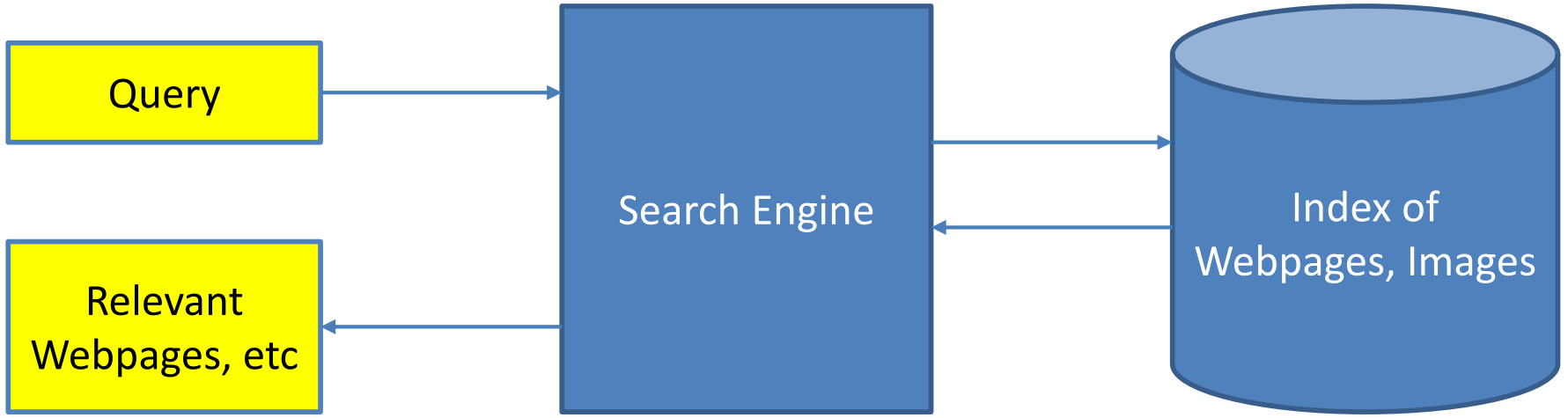
Slides: http://comp.nus.edu.sg/~xiangnan/sigir18-deep.pdf

Overview of Search Engine

Information pull: a user pulls information by making a specific request

User intent is explicitly reflected in query:
- Keywords, questions

Content is in
- Webpages, images, …



Key challenge: query-document semantic gap

Example of Query-Document Mismatch

Query	Document	Term matching	Semantic matching
seattle best hotel	seattle best hotels	partial	yes
pool schedule	swimming pool schedule	partial	yes
natural logarithm transformation	logarithm transformation	partial	yes
china kong	china hong kong	partial	no
why are windows so expensive	why are macs so expensive	partial	no

Same Search Intent Different Query Representations Example: “Distance between Sun and Earth”

“how far” earth sun	average distance from the earth to the sun
“how far” sun	how far away is the sun from earth
average distance earth sun	average distance from earth to sun
how far from earth to sun	distance from earth to the sun
distance from sun to earth	distance between earth and the sun
distance between earth & sun	distance between earth and sun
how far earth is from the sun	distance from the earth to the sun
distance between earth sun	distance from the sun to the earth
distance of earth from sun	distance from the sun to earth
“how far” sun earth	how far away is the sun from the earth
how far earth from sun	distance between sun and earth
how far from earth is the sun	how far from the earth to the sun
distance from sun to the earth	

Same Search Intent Different Query Representations Example: “Youtube”

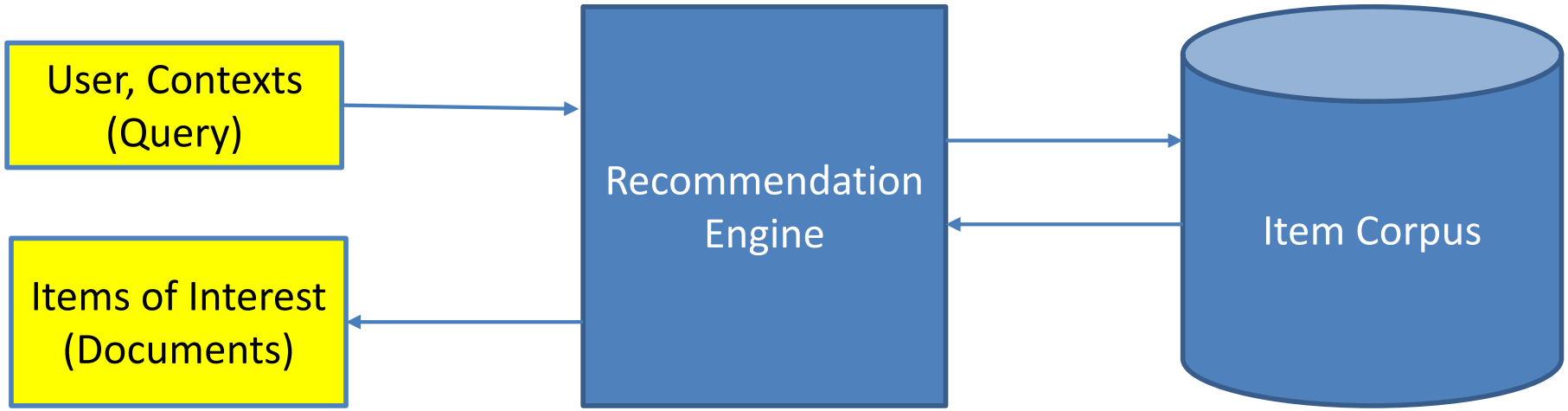
yutube	yuotube	yuo tube
ytube	youtubr	yu tube
youtubo	youtuber	youtubecom
youtube om	youtube music videos	youtube videos
youtube	youtube com	youtube co
youtub com	you tube music videos	yout tube
youtub	you tube com yourtube	your tube
you tube	you tub	you tube video clips
you tube videos	www you tube com	wwww youtube com
www youtube	www youtube com	www youtube co
yotube	www you tube	www utube com
ww youtube com	www utube	www u tube
utube videos	utube com	utube
u tube com	utub	u tube videos
u tube	my tube	toutube
outube	our tube	toutube

Overview of Recommendation Engine

Information push: the system pushes information to a user by guessing the user interest

User Interest is implicitly reflected in:
- Interaction history
- Demographics
- Contexts

Items can be:
- Products, news, movies, videos, friends …



Key challenge: user-item semantic gap

- Even severe than search, since user and item are two **different types of entities** and are represented by different features

Example of User-Item Semantic Gap

Movie Recommendation

?

User Profile (query):

- User ID
- Rating history
- Age, gender
- Income level
- Time of the day

…….

Item Profile (document):

- Item ID
- Description
- Category
- Price
- Image

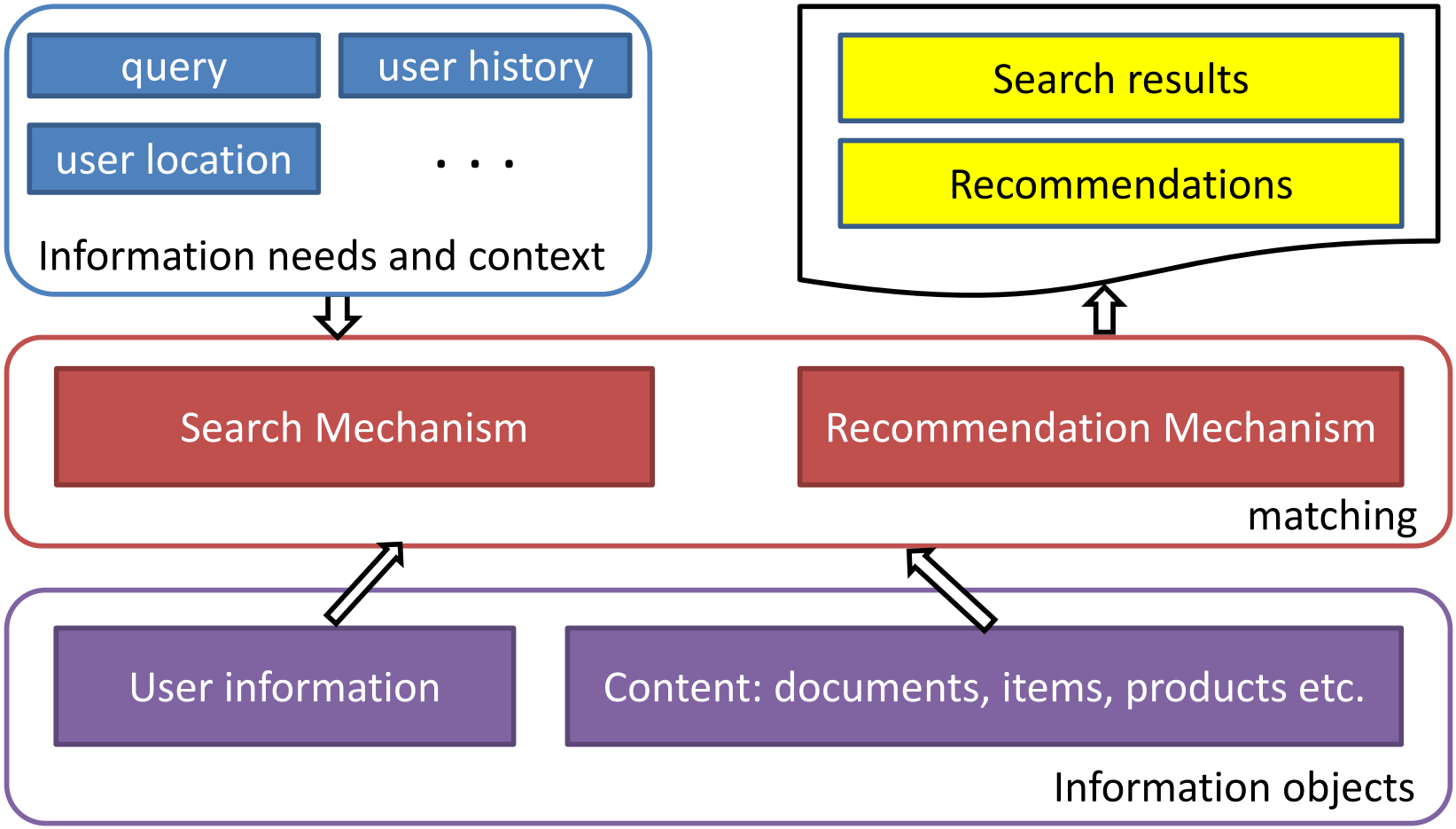
…….

There may be no overlap between user features and item features
Matching cannot be done on the superficial feature level!

Information Providing Mechanisms of Search and Recommendation (Hector et al., CACM' 11)

	Search	Recommendation
Delivery model	Pull	Push or pull
Beneficiary (priority)	User	User and provider
Unexpected good?	No	Yes
Collective knowledge	Maybe	Maybe
Query available	Yes	Maybe
Context dependent	Maybe	Maybe

Unified View on Matching in Search and Recommendation (Hector et al, CACM'11)



Common goal: matching a context (may or may not include an explicit query) to a collection of information objects (product descriptions, web pages, etc.)

Difference for search and recommendation: features used for matching!

Semantic Gap is Biggest Challenge in both Search and Recommendation

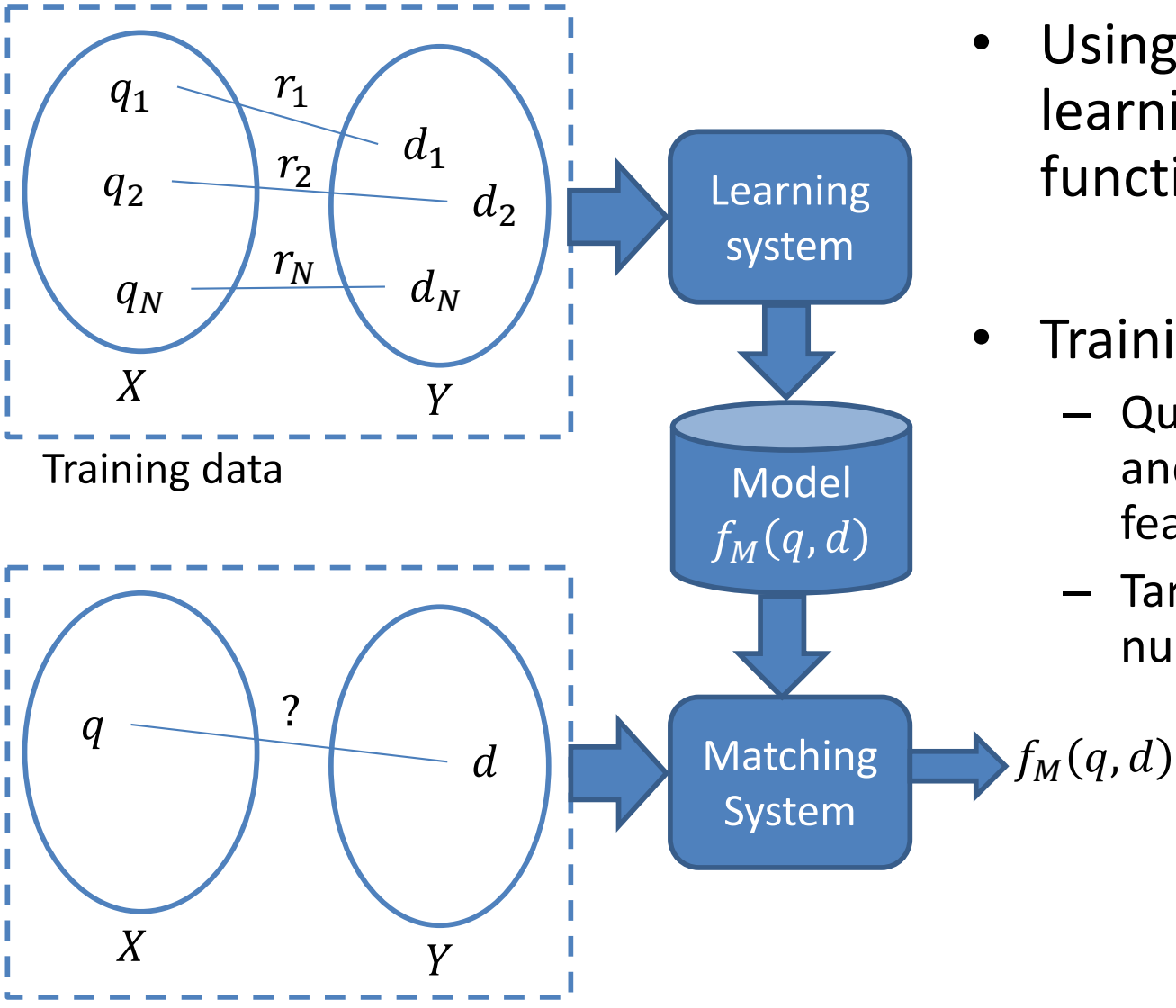
Query-document Mismatch

- Same intent can be represented by different queries (representations)
- Search is still mainly based on term level matching
- Query document mismatch occurs, when searcher and author use different representations

User-item Semantic Gap

- Features are used to represent a user and an item may be totally different (e.g., ID feature)
- Even when they partially overlap in features, it is insensible to conduct direct matching

Machine Learning for Matching



- Using relations in data for learning the matching function $f_M(q,d)$ or $P(r|q,d)$
- Training data $\{(q_i, d_i, r_i)\}_{i=1}^{N}$
  - Queries and documents (users and items) represented with feature vectors or ID's
  - Target can be binary or numerical values

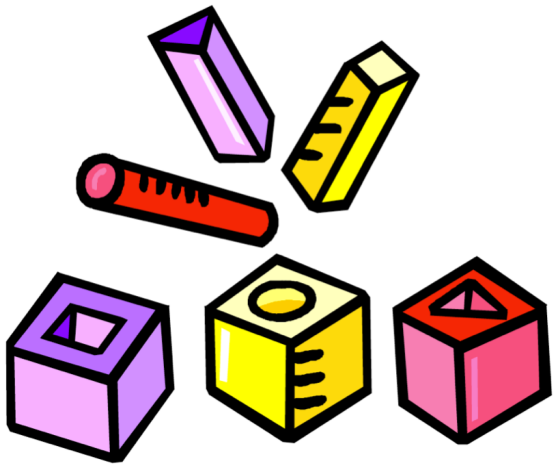
Organization of the Tutorial

- Unified View of Matching in Search and Recommendation (Jun Xu)
- Part 1: Traditional Approaches to Matching
  - Traditional matching models for search (Jun Xu)
  - Traditional matching models for recommendation (Xiangnan He)
- Part 2: Deep Learning Approaches to Matching
  - Overview (Jun Xu)
  - Deep matching models for search (Jun Xu)
  - Deep matching models for recommendation (Xiangnan He)
- Summary (Xiangnan He)

Outline of Tutorial

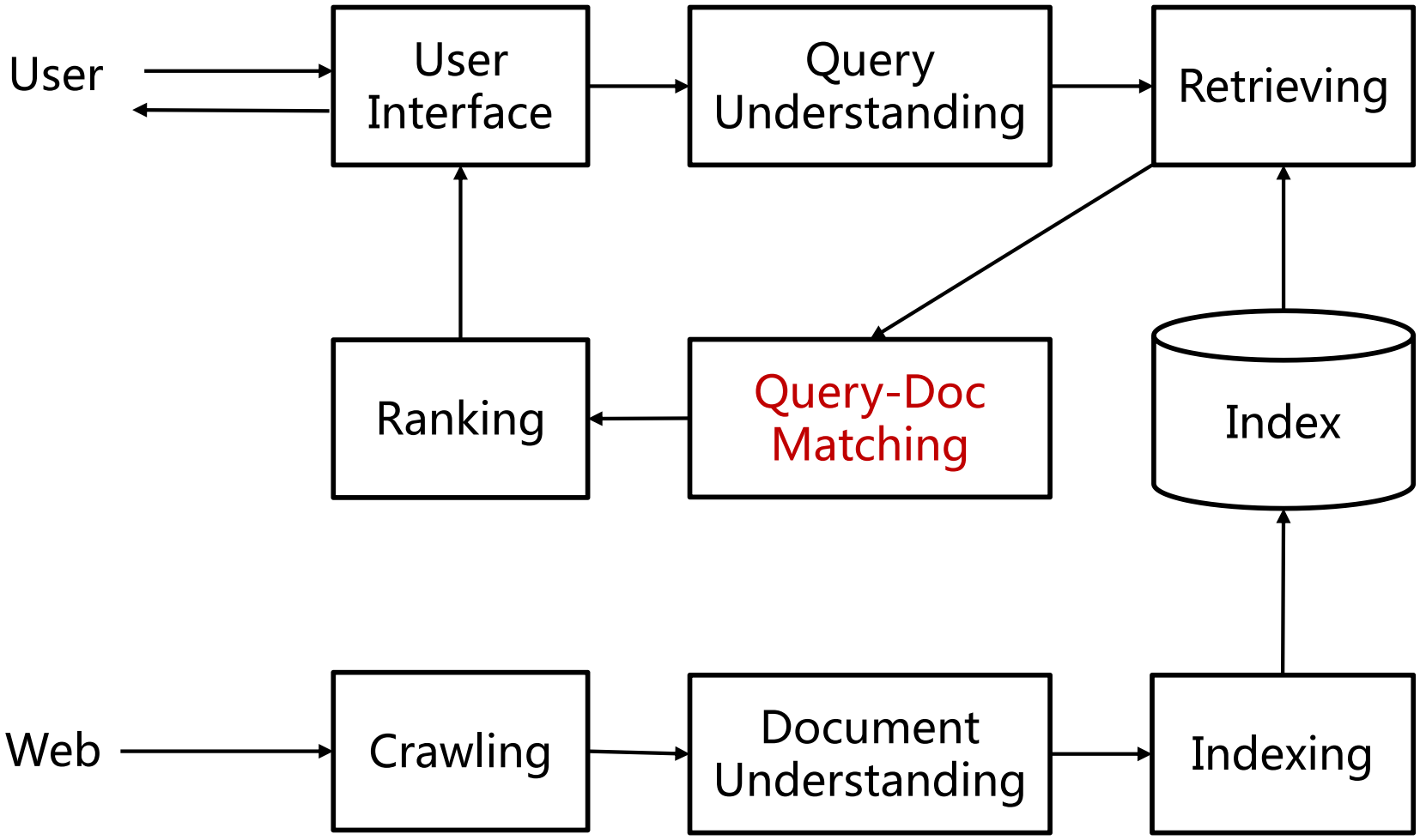
- Unified View of Matching in Search and Recommendation
- Part 1: Traditional Approaches to Matching
  - Traditional matching models for search
  - Traditional matching models for recommendation
- Part 2: Deep Learning Approaches to Matching
- Summary

Slides: http://comp.nus.edu.sg/~xiangnan/sigir18-deep.pdf



QUERY-DOCUMENT MATCHING

Overview of Web Search Engine



Key Factors for Query-Document Matching

Query:
Down the ages noodles and dumplings were famous Chinese food

Document:

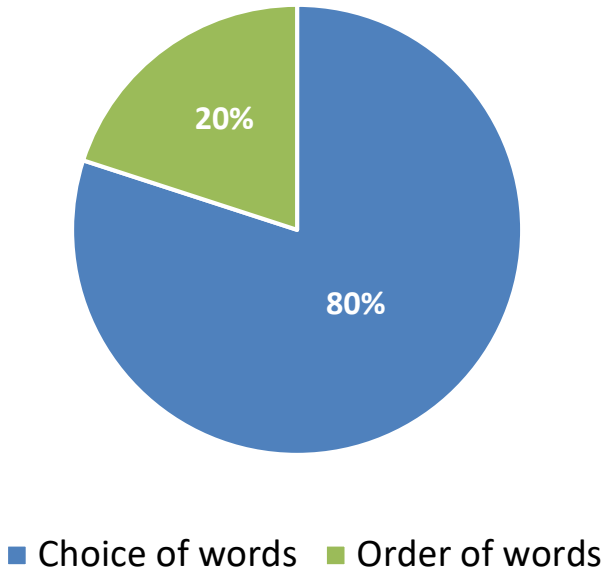
... down the ages dumplings and noodles were popular in China …

- Bridging the semantic gap between words
  - Semantically similar words: famous ~ popular, Chinese ~ China
- Capturing order of words
  - N-grams: “down the ages” ~ “down the ages”
  - N-terms: “noodles and dumplings” ~ “dumplings and noodles”

  ……

Information from Choice of Words and Order of Words (Ross, ’02)

- Assume:
  - Size of vocabulary = 10,000
  - Average sentence length = 20
- Rough contributions of information in bits
  - From the selection of words: $\log_2(10000^{20})$
  - From the order of words: $\log_2(20!)$
- “Over 80% of the potential information in language being in the choice of words without regard to the order in which they appear ”
  - 80%: choice of words
  - 20%: order of words



Relation between Matching and Ranking

- In traditional IR: ranking = matching

  $f(q,d) = f_{BM25}(q,d)$ or $f(q,d) = f_{LMIR}(d|q)$

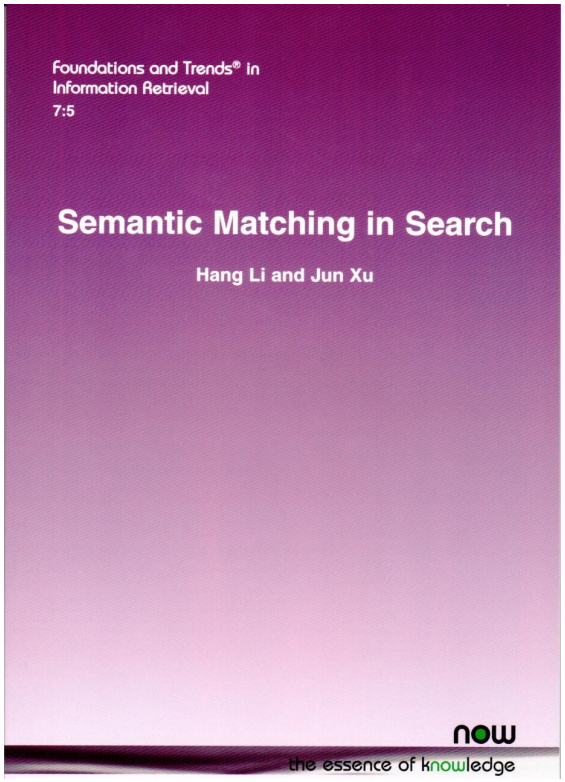
- In Web search: ranking and matching become separated
  - Learning to rank: matching as features for ranking

    $f(q,d) = f_{BM25}(q,d) + \text{PageRank}(d) + \cdots$

	Matching	Ranking
Prediction	Matching degree between a query and a document	Ranking list of documents
Model	$f(q,d)$	$f(q,\{d_1,d_2,\cdots\})$
Challenge	Mismatch	Correct ranking on the top

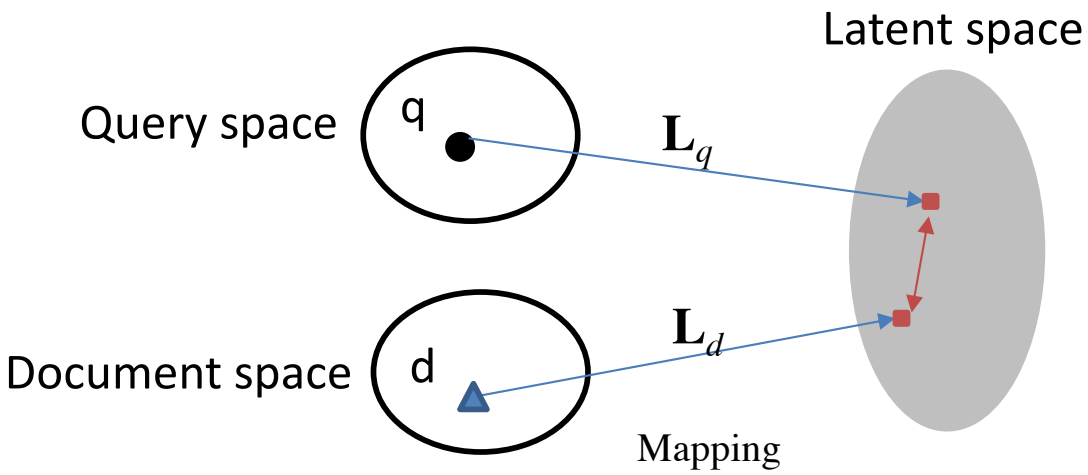
Traditional Approaches to Query-Document Semantic Matching

- Matching by query formulation
- Matching with term dependency
- Matching with topic model
- Matching in latent space model
- Matching with translation model

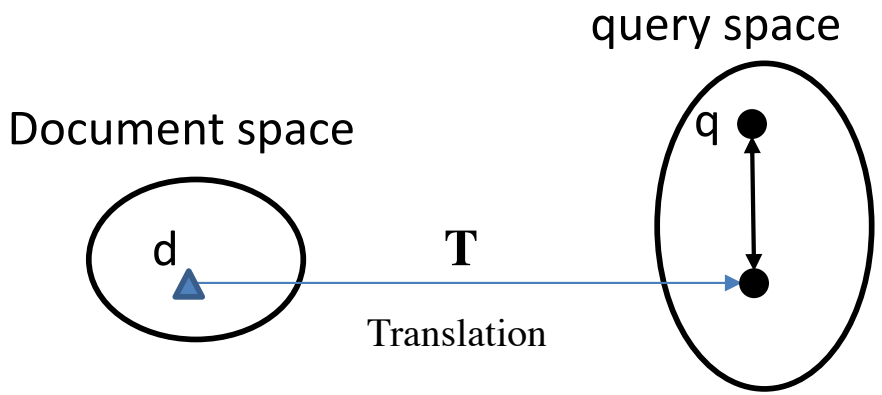


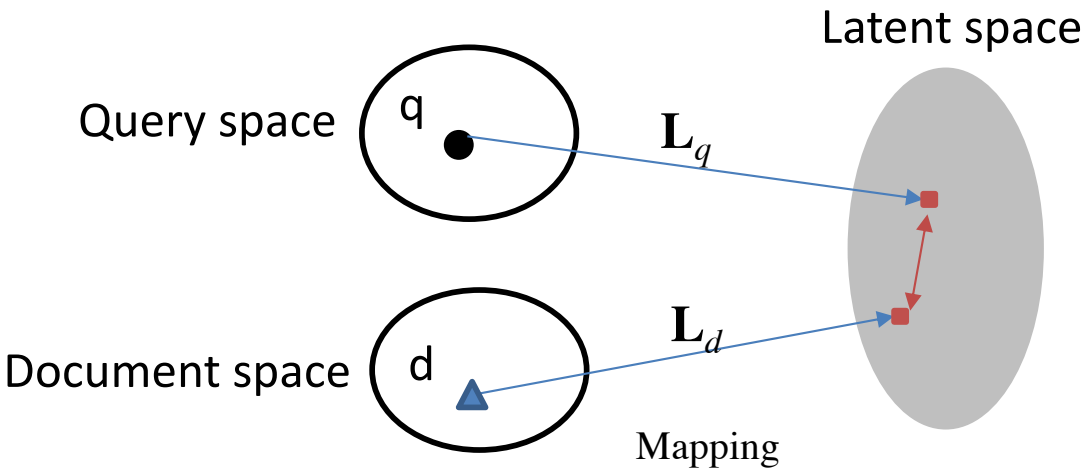
Traditional Matching Models for Search

- **Matching in latent space**: mapping query and document into a latent space



- **Matching with machine translation**: mapping document to query space





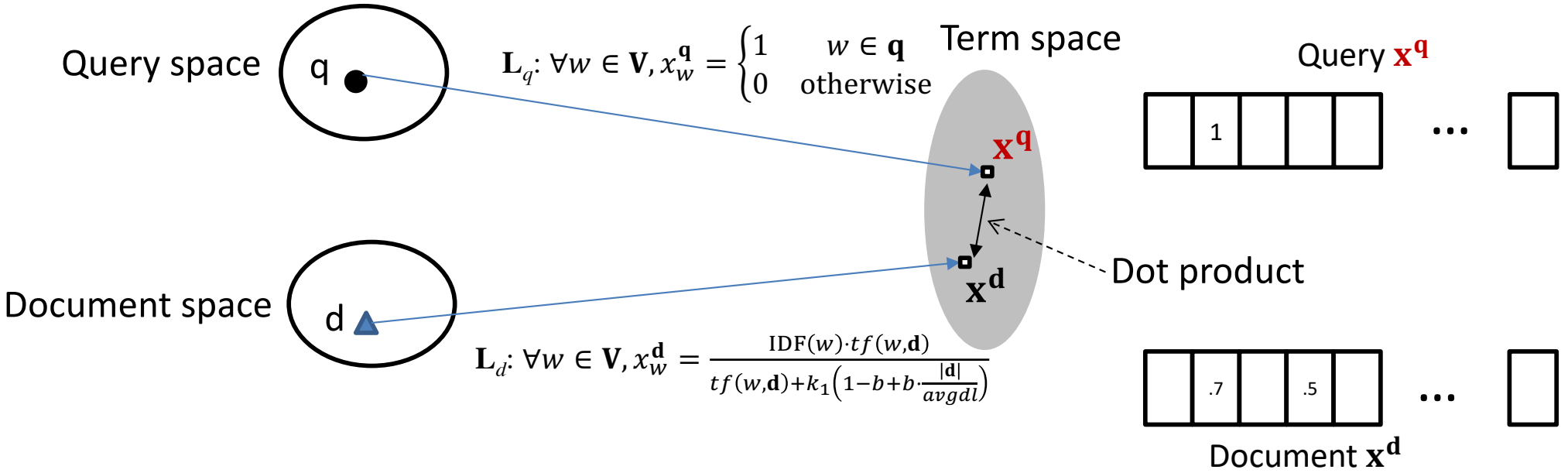
MATCHING IN LATENT SPACE

BM25: Matching in Term Space

$$f_{BM25}(\mathbf{q},\mathbf{d}) = \sum_{q_i \in \mathbf{q}} \mathrm{IDF}(q_i) \frac{tf(q_i,\mathbf{d})}{tf(q_i,\mathbf{d}) + k_1\left(1-b+b\cdot\frac{|\mathbf{d}|}{avgdl}\right)}$$

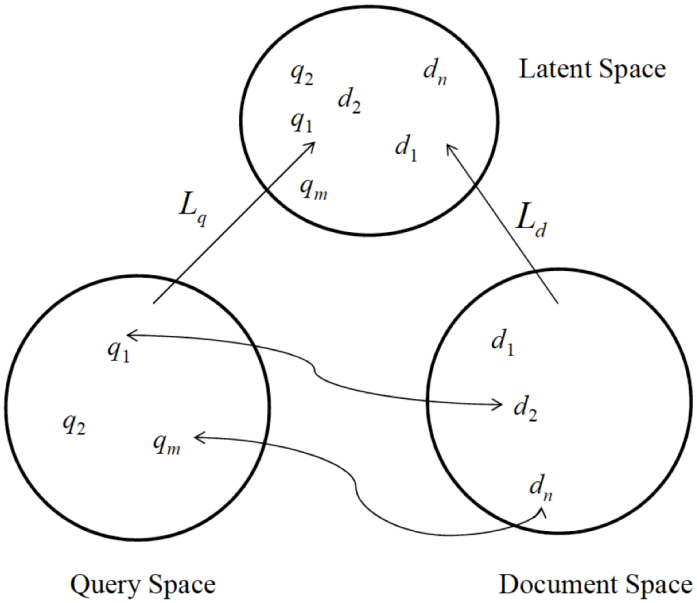
$$= \sum_{w \in \mathbf{V}} 1_{w \in \mathbf{q}} \times \frac{\mathrm{IDF}(w)\cdot tf(w,\mathbf{d})}{tf(w,\mathbf{d}) + k_1\left(1-b+b\cdot\frac{|\mathbf{d}|}{avgdl}\right)}$$

$$= \langle \mathbf{x}^{\mathbf{q}}, \mathbf{x}^{\mathbf{d}} \rangle$$



Matching in Latent Space

- Assumption
  - Queries/documents have similarities
  - Click-through data represent “similarity” relations between queries and documents
- Approach
  - Project queries and documents to latent space
  - With some regularization or constraints



Partial Least Square (PLS)

- Two spaces: $\mathcal{X} \subset \mathbb{R}^m$ and $\mathcal{Y} \subset \mathbb{R}^n$
- Training data: $\{(x_i, y_i, r_i)\}_{i=1}^N$, $r_i \in \{+1, -1\}$ or $r_i \in \mathbb{R}$
- Model
  - Dot product as similarity: $f(x, y) = \langle L_X^T x, L_Y^T y \rangle = x^T L_X L_Y^T y$
  - L_X and L_Y are two linear (and orthonormal) transformations
- Objective function

$$\text{argmax}_{L_X, L_Y} \sum_{r_i=+1} x_i^T L_X L_Y^T y_i - \sum_{r_i=-1} x_i^T L_X L_Y^T y_i$$

$$\text{s.t. } L_X^T L_X = I_{K\times K}, L_Y^T L_Y = I_{K\times K}$$

Regularized Mapping to Latent Space (RMLS)

- Two spaces: $\mathcal{X} \subset \mathbb{R}^m$ and $\mathcal{Y} \subset \mathbb{R}^n$
- Training data: $\{(x_i, y_i, r_i)\}_{i=1}^N$, $r_i \in \{+1, -1\}$ or $r_i \in \mathbb{R}$
- Model
  - Dot product as similarity: $f(x, y) = \langle L_X^T x, L_Y^T y \rangle = x^T L_X L_Y^T y$
  - L_X and L_Y are two linear transformations with ℓ_1 and ℓ_2 regularizations (sparse transformations)
- Objective function

$$\text{argmax}_{L_X, L_Y} \sum_{r_i=+1} x_i^T L_X L_Y^T y_i - \sum_{r_i=-1} x_i^T L_X L_Y^T y_i$$

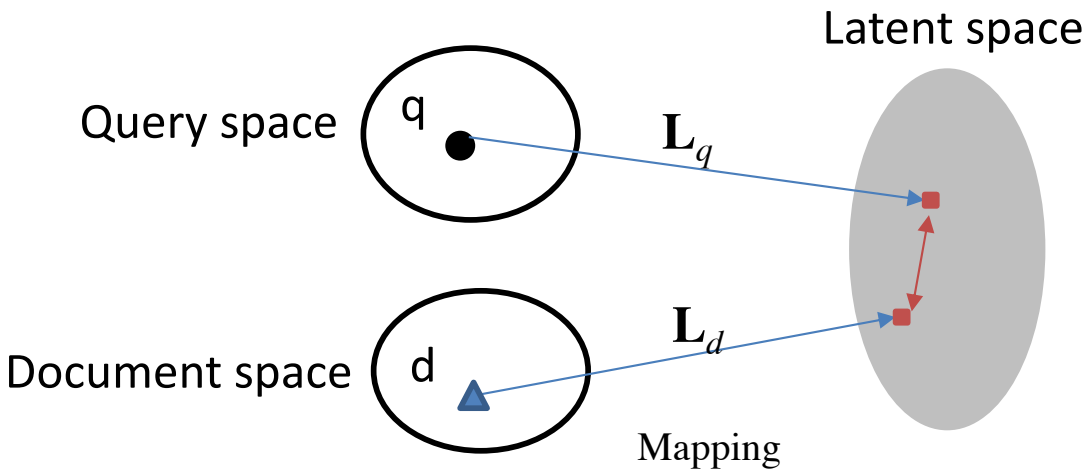
$$\text{s.t. } |L_X| \leq \lambda_X, |L_Y| \leq \lambda_Y, \|L_X\| \leq \vartheta_X, \|L_X\| \leq \vartheta_Y$$

PLS v.s. RMLS

	PLS	RMLS
Transformation Assumption	orthonormal	L1 and L2 regularization
Optimization Method	singular value decomposition	coordinate ascent
Optimality	global optimum	local optimum
Efficiency	low	high
Scalability	low	high

Bridging the Semantic Gap

- Latent space models bridge semantic gap between words through
  - Reducing the dimensionality (from term level matching to semantic matching)
  - Correlating semantically similar terms (matrices are not diagonal)
- Automatically learning mapping functions from data

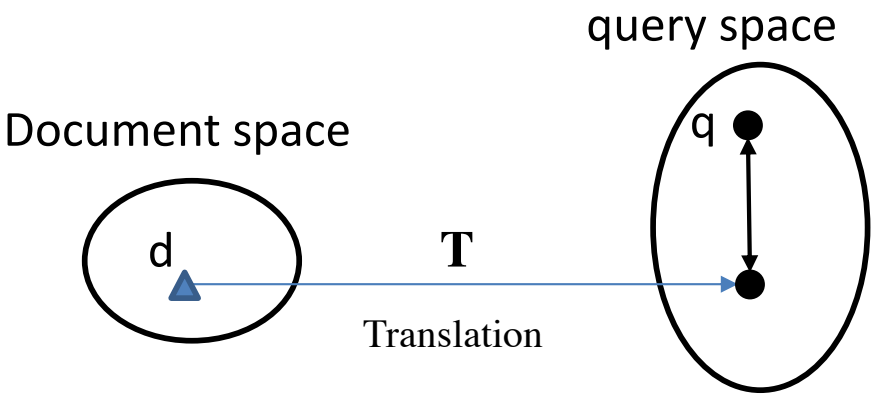


Experimental Results

	NDCG@1	NDCG@3	NDCG@5
BM25	0.637	0.690	0.690
SSI	0.538	0.621	0.629
BLTM	0.657	0.702	0.701
PLS	0.676	0.728	**0.736**
RMLS	**0.686**	**0.732**	0.729

Based on a web search data set containing 94,022 queries and 111,631 documents. Click through associated with the queries and documents at a search engine is used.

- Latent space models work better than baseline (BM25)
- RMLS works equally well as PLS, with higher learning efficiency and scalability



MATCHING WITH TRANSLATION MODEL

Statistical Machine Translation (SMT)

- Given a sentence C in source language, translates it into sentence E in target language

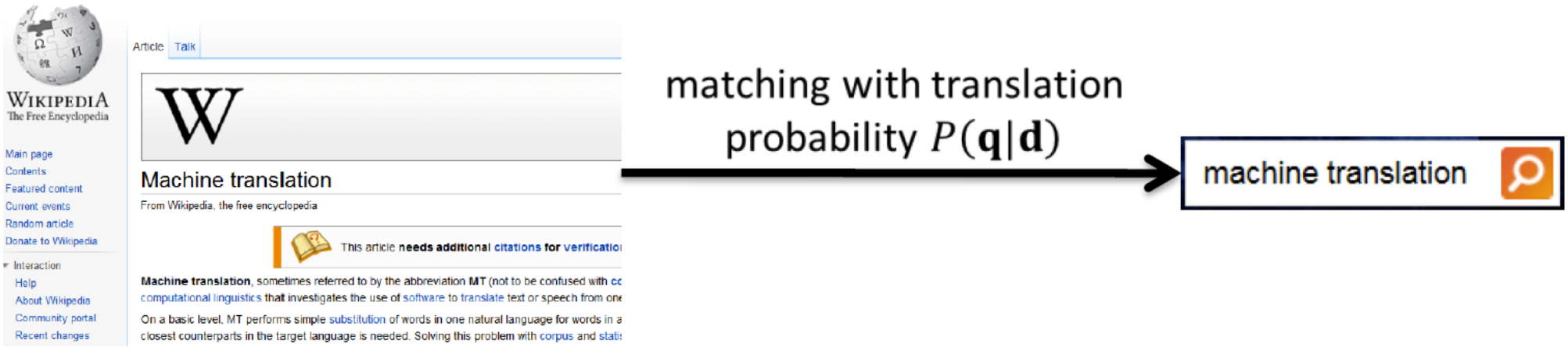
$$E^* = \text{argmax}_E P(E|C)$$

- Linear combination of features

$$P(E|C) = \frac{1}{Z(C,E)} \exp \sum_i \lambda_i h(C,E)$$

$$E^* = \text{argmax}_E \sum_i \lambda_i h(C,E)$$

Statistical Machine Translation for Query-Document Matching



- Translating document **d** to query **q**
- Matching degree: translation probability $P(\mathbf{q}|\mathbf{d})$
- Key difference from conventional translation model
  - Translation within the same language (need to handle self-translation)

Matching with Word-based Translation Models

- Basic model

$$P(\mathbf{q}|\mathbf{d}) = \prod_{q \in \mathbf{q}} P(q|\mathbf{d}) = \prod_{q \in \mathbf{q}} \sum_{w \in \mathbf{d}} P(q|w)P(w|\mathbf{d})$$

translation probability

Document language model

- Smoothing to avoid zero translation probability (Berger & Lafferty '99)

$$P(\mathbf{q}|\mathbf{d}) = \prod_{q \in \mathbf{q}} \left(\alpha P(q|C) + (1-\alpha) \left(\sum_{w \in \mathbf{d}} P(q|w)P(w|\mathbf{d}) \right) \right)$$

background language model

- Self-translation (Gao et al., '10)

$$P(\mathbf{q}|\mathbf{d}) = \prod_{q \in \mathbf{q}} \left(\alpha P(q|C) + (1-\alpha) \left(\beta P(q|\mathbf{d}) + (1-\beta) \left(\sum_{w \in \mathbf{d}} P(q|w)P(w|\mathbf{d}) \right) \right) \right)$$

unsmoothed document language model

Bridging Semantic Gap between Words

- Translation matrix can bridge semantic gap between query words and document words

q	$t(q \mid w)$
solzhenitsyn	0.319
citizenship	0.049
exile	0.044
archipelago	0.030
alexander	0.025
soviet	0.023
union	0.018
komsomolskaya	0.017
treason	0.015
vishnevskaya	0.015

$w =$ solzhenitsyn

q	$t(q \mid w)$
carcinogen	0.667
cancer	0.032
scientific	0.024
science	0.014
environment	0.013
chemical	0.012
exposure	0.012
pesticide	0.010
agent	0.009
protect	0.008

$w =$ carcinogen

q	$t(q \mid w)$
zubin_mehta	0.248
zubin	0.139
mehta	0.134
philharmonic	0.103
orchestra	0.046
music	0.036
bernstein	0.029
york	0.026
end	0.018
sir	0.016

$w =$ zubin

q	$t(q \mid w)$
pontiff	0.502
pope	0.169
paul	0.065
john	0.035
vatican	0.033
ii	0.028
visit	0.017
papal	0.010
church	0.005
flight	0.004

$w =$ pontiff

q	$t(q \mid w)$
everest	0.439
climb	0.057
climber	0.045
whittaker	0.039
expedition	0.036
float	0.024
mountain	0.024
summit	0.021
highest	0.018
reach	0.015

$w =$ everest

q	$t(q \mid w)$
wildlife	0.705
fish	0.038
acre	0.012
species	0.010
forest	0.010
environment	0.009
habitat	0.008
endangered	0.007
protected	0.007
bird	0.007

$w =$ wildlife

From Berger & Lafferty, '99

Experimental Results

Models	NDCG@1	NDCG@3	NDCG@10
BM25 (baseline)	0.3181	0.3413	0.4045
WTM (without self-translation)	0.3210	0.3512	0.4211
WTM (with self-translation)	0.3310	0.3566	0.4232

Based on a large scale real world data set containing 12,071 English queries sampled from one-year query log files of a commercial search engine (Gao et al., 2010)

- Word-based translation model (WTM) outperformed the baseline
  - Translation probabilities bridge the semantic gap between query words and document words
  - Self-translation is effective

References

- Adam Berger, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu Mittal. Bridging the Lexical Chasm: Statistical Approaches to Answer-Finding. In SIGIR 2000.
- Jianfeng Gao, Xiaodong He, and JianYun Nie. Click-through-based Translation Models for Web Search: from Word Models to Phrase Models. In CIKM 2010.
- Jianfeng Gao, Kristina Toutanova, and Wen-tau Yih. Clickthrough-based latent semantic models for web search. In SIGIR 2011.
- Jianfeng Gao : Statistical Translation and Web Search Ranking. http://research.microsoft.com/en-us/um/people/jfgao/paper/SMT4IR.res.pptx
- Dustin Hillard, Stefan Schroedl, and Eren Manavoglu, Hema Raghavan, and Chris Leggetter. Imrpoved Ad Relevance in Sponsored Search. In WSDM 2010.
- Jian Huang, Jianfeng Gao, Jiangbo Miao, Xiaolong Li, Kuansan Wang, Fritz Behr, and C. Lee Giles. Exploring web scale language models for search query processing. In WWW 2010.
- Ea-Ee Jan, Shih-Hsiang Lin, and Berlin Chen. Translation Retrieval Model for Cross Lingual Information Retrieval. In AIRS 2010.
- Rong Jin, Alex G. Hauptmann, and Chengxiang Zhai. Title Language Model for Information Retrieval. In SIGIR 2002.
- Maryan Karimzadehgan and Chengxiang Zhai. Estimation of Statistical Translation Models based on Mutual Information for Ad Hoc Information Retrieval. In SIGIR 2010.
- David Mimno , Hanna M. Wallach , Jason Naradowsky , David A. Smith, Andrew McCallum. Polylingual topic models. In EMNLP 2009.

References

- Adam Berger and John Lafferty. Information Retrieval as Statistical Translation. In SIGIR 1999.
- Jae-Hyun Park, W. Bruce Croft, and David A. Smith. Qusi-Synchronous Dependence Model for Information Retrieval. In CIKM 2011.
- Stefan Riezler and Yi Liu. Query Rewritting Using Monolingual Statistical Machine Translation. In ACL 2010.
- Dolf Trieschnigg, Djoerd Hiemstra, Franciska de Jong, and Wessel Kraaij. A cross-lingual Framework for Monolingual Biomedical Information Retrieval. In CIKM 2010.
- Elisabeth Wolf, Delphine Bernhard, and Iryan Gurevych. Combining Probabilistic and Translation-based Models for Information Retrieval based on Word Sense Annotations. In CLEF Workshop 2009.
- D.R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. Neural Computation, 2004.
- Jianfeng Gao, Kristina Toutanova and Wen-tau Yih. Clickthrough-based latent semantic models for web search. In Proc. of SIGIR, 2011.
- R. Rosipal and N. Krämer. Overview and recent advances in partial least squares. Subspace, Latent Structure and Feature Selection, 2006.
- Wei Wu, Hang Li, and Jun Xu. Learning Query and Document Similarities from Click-through Bipartite Graph with Metadata. Microsoft Research Technical Report, 2011.
- Jun Xu, Hang Li, Chaoliang Zhong, Relevance Ranking Using Kernels, In Proceedings of the 6th Asian Information Retrieval Societies Symposium (AIRS'10), 1-12, 2010.
- Hector Garcia-Molina, Georgia Koutrika, Aditya Parameswaran, Information Seeking: Convergence of Search, Recommendations, and Advertising Communications of the ACM, Vol. 54 No. 11, Pages 121-130.
- Brian H. Ross. Psychology of Learning and Motivation: Advances in Research and Theory. Elsevier. 2002.

Outline of Tutorial

- Unified View of Matching in Search and Recommendation
- Part 1: Traditional Approaches to Matching
  - Traditional matching models for search
  - Traditional matching models for recommendation
    - Collaborative Filtering Models
    - Generic Feature-based Models
- Part 2: Deep Learning Approaches to Matching
- Summary

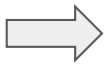
Slides: http://comp.nus.edu.sg/~xiangnan/sigir18-deep.pdf

Collaborative Filtering

- Collaborative Filtering (CF) is the most well-known technique for recommendation.
  - Homophily assumption: a user preference can be predicted from his/her similar users.
- Math formulation: matrix completion problem

User	Movie	Rating
Alice	Titanic	5
Alice	Notting Hill	3
Alice	Star Wars	1
Bob	Star Wars	4
Bob	Star Trek	5
Charlie	Titanic	1
Charlie	Star Wars	5
. . .	. . .	. . .

Input Tabular data



User \ Movie	TI	NH	SW	ST	...
A	5	3	1	?	...
B	?	?	4	5	...
C	1	?	5	?	...
...	...	...	...	...	...

Rating Matrix
(Interaction Matrix)

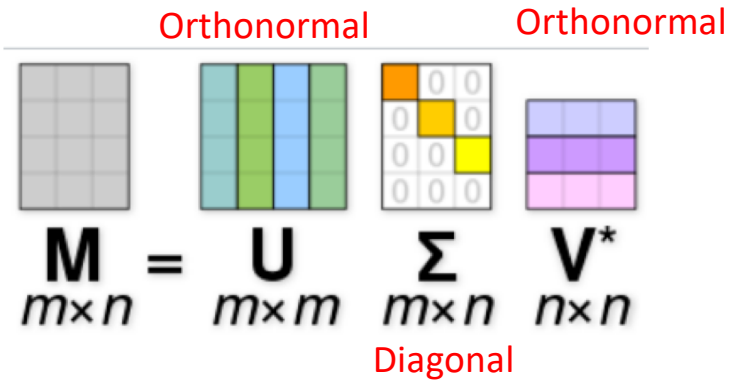
Solving Matrix Completion

- Singular Value Decomposition (SVD) is the most well-known technique for matrix completion

Movie

User	TI	NH	SW	ST	...
A	5	3	1	?	...
B	?	?	4	5	...
C	1	?	5	?	...
...	...	...	...	...	...

Rating Matrix

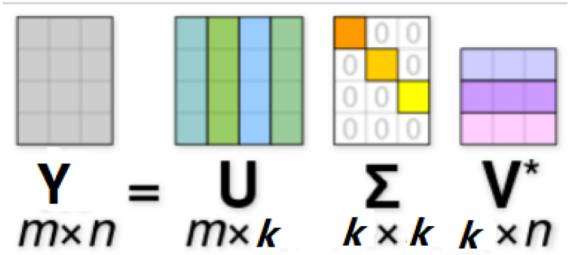


Steps to use SVD for CF:

1. Impute missing data to 0 in **Y**
2. Solving the SVD problem
3. Using only *K* dimensions in **U** and **V** to obtain a low rank model to estimate **Y**

Figure adopted from: https://en.wikipedia.org/wiki/Singular-value_decomposition

SVD is Suboptimal for CF



- In essence, SVD is solving the problem:

$$\arg\min_{\mathbf{U},\Sigma,\mathbf{V}} (\mathbf{Y} - \mathbf{U}\Sigma\mathbf{V}^T)^2$$

$$= \arg\min_{\mathbf{U},\Sigma,\mathbf{V}} \sum_{i=1}^{m}\sum_{j=1}^{n} (y_{ij} - (\mathbf{U}\Sigma\mathbf{V}^T)_{ij})^2$$

Label

Model Prediction

Training instance

- Several Implications (weaknesses):
  1. Missing data has the same weight as observed data (>99% sparsity)
  2. No regularization is enforced – easy to overfit

Adjust SVD for CF

- The "SVD" method the context of recommendation:
  - Model:

$$\hat{y}_{ui} = \mathbf{v}_u^T \mathbf{v}_i$$

User latent vector Item latent vector

  - Regularized Loss function:

$$L = \sum_u \sum_i w_{ui}(y_{ui} - \hat{y}_{ui})^2 + \lambda\left(\sum_u ||\mathbf{v}_u||^2 + \sum_i ||\mathbf{v}_i||^2\right)$$

Prediction error L2 regularizer

- This method is also called *Matrix Factorization* (MF) in RecSys:
  - It represents a user and an item as a latent vector (ID embedding).
  - The interaction between user and item is modeled using inner product

(measure how much user latent "preferences" match with item "properties"

  - Besides L2 loss, other loss can also be used, e.g., cross-entropy, margin-based pairwise loss, etc.

Factored Item Similarity Model (Kabbur et al., KDD'14)

- MF encodes a user with an ID, and projects it to embedding.
- Another more information-rich encoding is to use rated items of the user.
  - Also called as item-based CF (i.e., find similar items for recom)

$$\hat{y}_{ui} = \Big(\sum_{j \in \mathcal{R}_u} \mathbf{q}_j\Big)^T \mathbf{v}_i$$

⇔ user representation

Can be interpreted as the **similarity** between item i and j

Items rated by u

SVD++: Fusing User-based and Item-based CF (Koren, KDD'08)

- MF (user-based CF) represents a user as her ID.
  - Directly projecting the ID into latent space
- FISM (item-based CF) represents a user as her interacted items.
  - Projecting interacted items into latent space
- SVD++ fuses the two types of models in the latent space:

$$\hat{y}_{ui} = (\mathbf{v}_u + \sum_{j \in \mathcal{R}_u} \mathbf{q}_j)^T \mathbf{v}_i$$

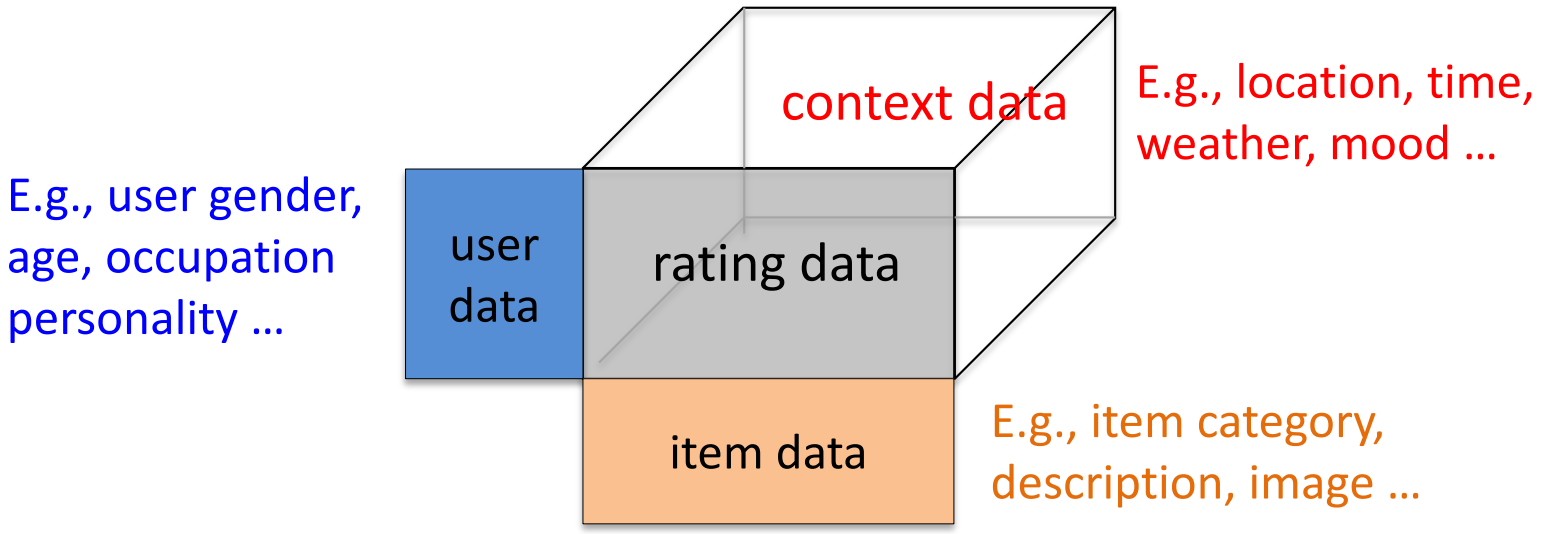
User representation in latent space

  - This is the best single model for rating prediction in the Netflix challenge (3 years, 1 million prize).

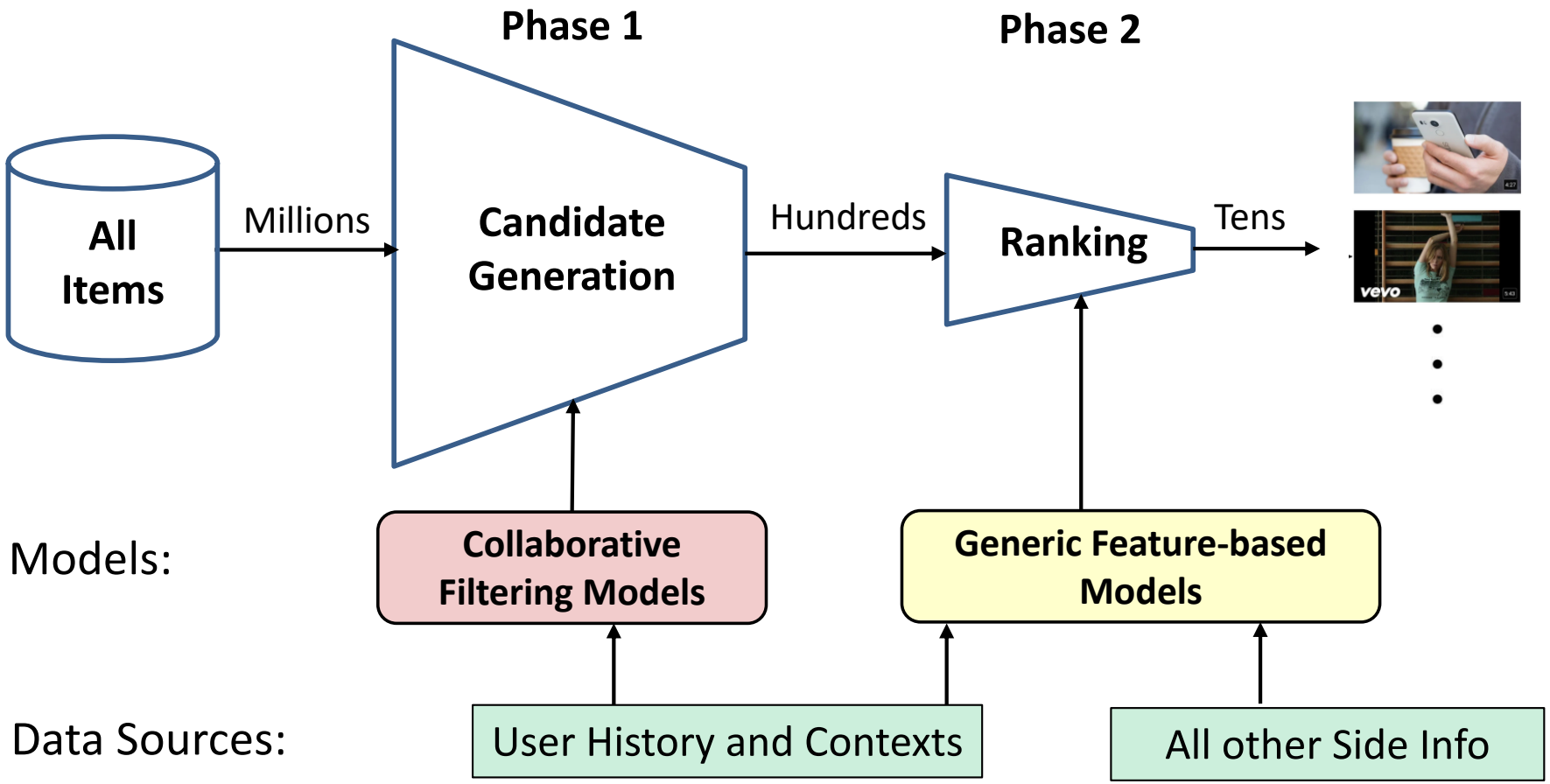
Note: the normalization terms are discarded for clarity.

How about Side Info?

- CF utilizes only the interaction matrix only to build the predictive model.
- How about other information like user/item attributes and contexts?
- Example data used for building a RecSys:

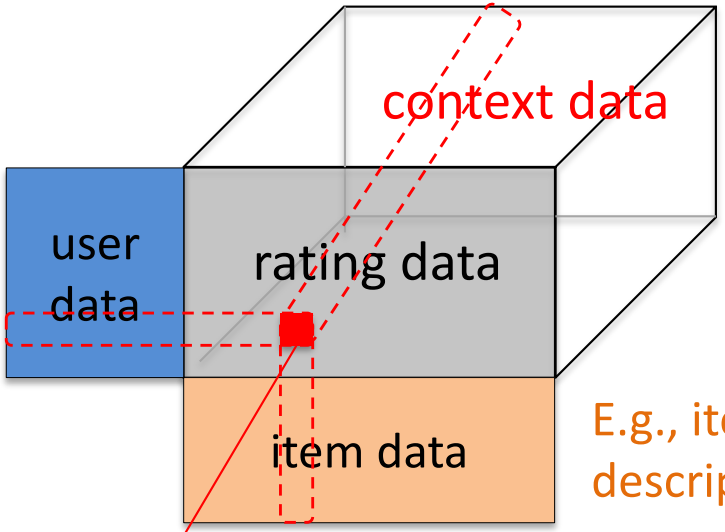


Modern RecSys Architecture



(Covington et al, Recsys'16)

Input to Feature-based Models



E.g., user gender, age, occupation personality …

E.g., recent history, location, time, weather, mood …

E.g., item category, description, image …

One-hot encoding

Each row encodes all info for a rating

Feature vector **x**

	A	B	C	…	TI	NH	SW	ST	…	TI	NH	SW	ST	…	Target y
	User				Movie					Other Movies rated					
$x^{(1)}$	1	0	0	…	1	0	0	0	…	0.3	0.3	0.3	0	…	5 $y^{(1)}$
$x^{(2)}$	1	0	0	…	0	1	0	0	…	0.3	0.3	0.3	0	…	3 $y^{(2)}$
$x^{(3)}$	1	0	0	…	0	0	1	0	…	0.3	0.3	0.3	0	…	1 $y^{(3)}$
$x^{(4)}$	0	1	0	…	0	0	1	0	…	0	0	0.5	0.5	…	4 $y^{(4)}$
$x^{(5)}$	0	1	0	…	0	0	0	1	…	0	0	0.5	0.5	…	5 $y^{(5)}$
$x^{(6)}$	0	0	1	…	1	0	0	0	…	0.5	0	0.5	0	…	1 $y^{(6)}$
$x^{(7)}$	0	0	1	…	0	0	1	0	…	0.5	0	0.5	0	…	5 $y^{(7)}$

Sparse Predictive Model

FM: Factorization Machines (Rendle, ICDM'10)

- It represents each feature an embedding vector, and models the second-order feature interactions:

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{p} w_i x_i + \sum_{i=1}^{p} \sum_{j>i}^{p} < \mathbf{v}_i, \mathbf{v}_j > x_i x_j$$

Only nonzero features are considered

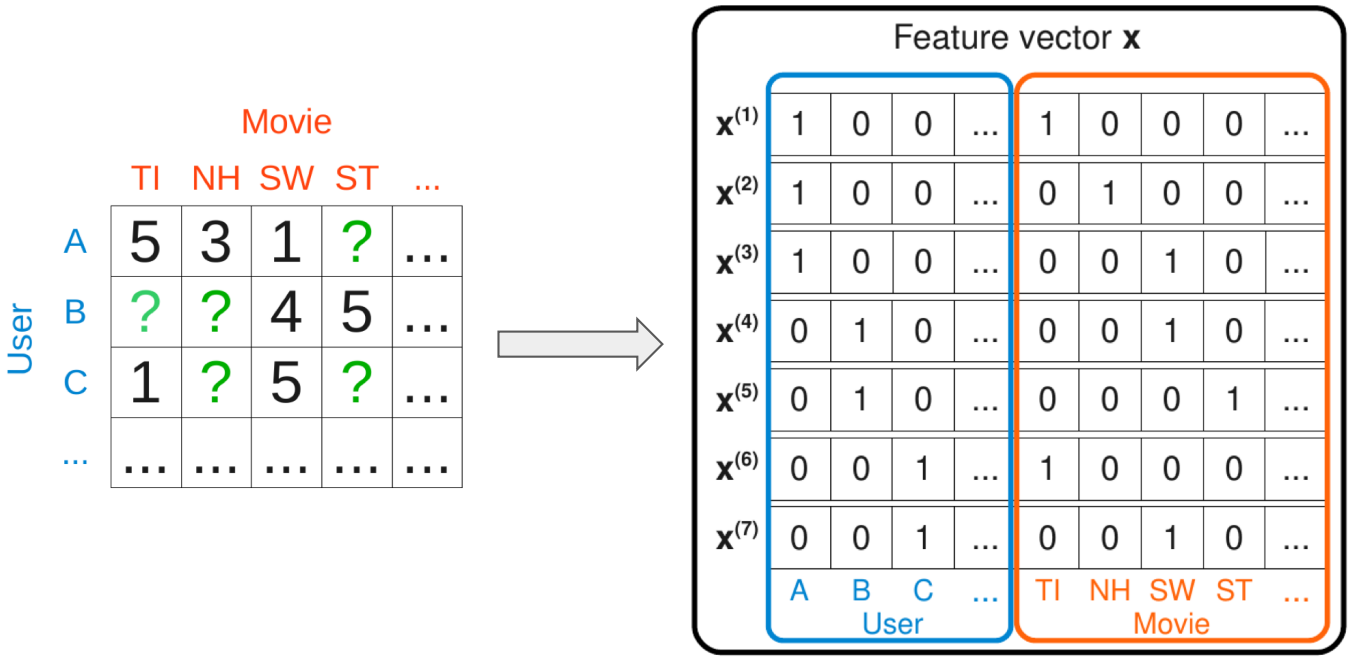
First-order: Linear Regression

Second-order: pair-wise interactions between features

  - Note: self-interaction is not included: $< \mathbf{v}_i, \mathbf{v}_i >$.
- FM allows easy feature engineering for recommendation, and can mimic many existing models (that are designed for a specific task) by inputting different features.
  - E.g., MF, SVD++, timeSVD (Koren, KDD'09), PITF (Rendle, WSDM'10) etc.

Matrix Factorization with FM

- Input: 2 variables <user (ID), item (ID)>.

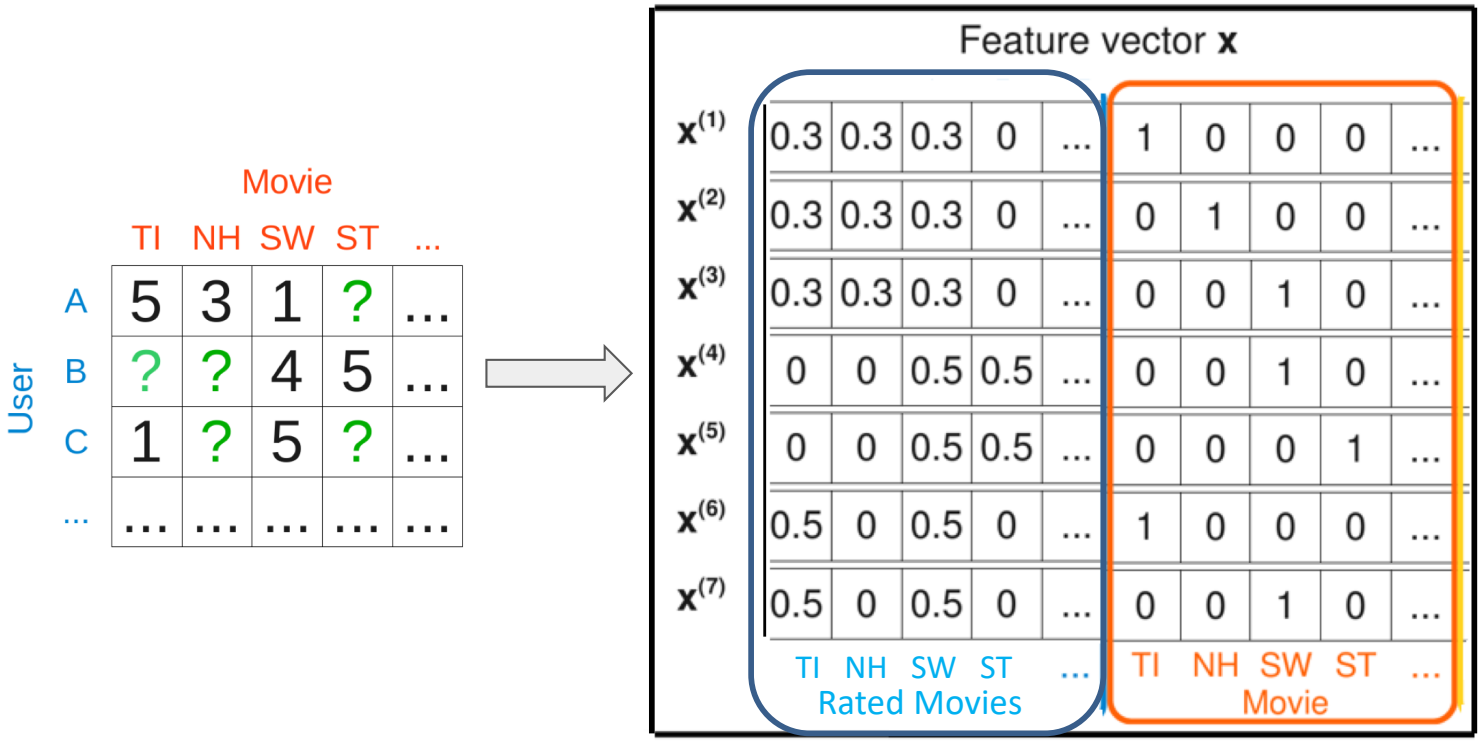


With this input, FM is identical to MF with bias:

$$\hat{y}(\mathbf{x}) = w_0 + w_u + w_i + \underbrace{< \mathbf{v}_u, \mathbf{v}_i >}_{\text{MF}}$$

Factored Item Similarity Model with FM

- Input: 2 variables <user (historical items ID), item (ID)>.



Further input user ID into FM will resume SVD++

With this input, FM subsumes FISM with additional terms:

$$\hat{y}(\mathbf{x}) = bias + \underbrace{\sum_{j \in \mathcal{R}_u} < \mathbf{v}_j, \mathbf{v}_i >}_{\text{FISM}} + \sum_{j \in \mathcal{R}_u, j' > j} < \mathbf{v}_j, \mathbf{v}_{j'} >$$

Next: Learning Recommender Models
Rating Prediction is Suboptimal

- Old work on recommendation optimize L2 loss:

$$L = \sum_u \sum_i w_{ui}(y_{ui} - \hat{y}_{ui})^2 + \lambda(\sum_u ||\mathbf{v}_u||^2 + \sum_i ||\mathbf{v}_i||^2)$$

  - But many empirical evidence show that:

    A lower error rate does not lead to a good ranking performance…

  - Possible Reasons:

    1) Discrepancy between error measure (e.g., RMSE) and ranking measure.
    2) Observation bias – users tend to rate on the items they like.

Towards Top-N Recommendation

- Recommendation is a personalized ranking task by nature, rather than rating prediction (regression).
  - Evaluated by Precision/Reall/AUC etc, rather than RMSE!
- Optimizing the relative ranking of a user on two items are more advantageous:
  - Higher rating > Lower rating (explicit feedback)
  - Observed interaction > Unobserved interaction (implicit feedback)

sigmoid Positive prediction Negative prediction

$$L_{BPR} = \arg\max_{\Theta} \sum_{(u,i,j) \in \mathcal{R}_B} \ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) - \lambda ||\Theta||^2$$

Pairwise training examples: u prefers i over j

- Known as the Bayesian Personalized Ranking objective (BPR, Rendle et al, UAI'09)

References

- https://en.wikipedia.org/wiki/Collaborative_filtering
- Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. Fast matrix factorization for online recommendation with implicit feedback. In SIGIR 2016.
- Yehuda Koren, and Robert Bell. Advances in collaborative filtering. Recommender systems handbook. Springer, Boston, MA, 2015. 77-118.
- Santosh Kabbur, Xia Ning, and George Karypis. Fism: factored item similarity models for top-n recommender systems. In KDD 2013.
- Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In KDD 2018.
- Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In Recsys 2016.
- Steffen Rendle. Factorization machines. In ICDM 2010.
- Yehuda Koren. Collaborative filtering with temporal dynamics. Communications of the ACM 53, no. 4 (2010): 89-97.
- Steffen Rendle, and Lars Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In WSDM 2010.
- Immanuel Bayer, Xiangnan He, Bhargav Kanagal, and Steffen Rendle. A generic coordinate descent framework for learning from implicit feedback. In WWW 2017.
- Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In WWW 2017.
- Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In UAI 2009.

Outline of Tutorial

- Unified View of Matching in Search and Recommendation
- Part 1: Traditional Approaches to Matching
- Part 2: Deep Learning Approaches to Matching
  - Overview
  - Deep matching models for search
  - Deep matching models for recommendation
- Summary

Slides: http://comp.nus.edu.sg/~xiangnan/sigir18-deep.pdf

Growing Interests in "Deep Matching"

- Success of deep learning in other fields
  - Speech recognition, computer vision, and natural language processing
- Growing presence of deep learning in IR research
  - SIGIR keynote, Tutorial, and Neu-IR workshop
- Adopted by industry
  - ACM News: *Google Turning its Lucrative Web Search Over to AI Machines* (Oct. 26, 2015)
  - WIRED: *AI is Transforming Google Search. The Rest of the Web is Next* (April 2, 2016)
- Chris Manning (Stanford)'s SIGIR 2016 keynote:
  "I'm certain that *deep learning will come to dominate SIGIR over the next couple of years* … just like speech, vision, and NLP before it."

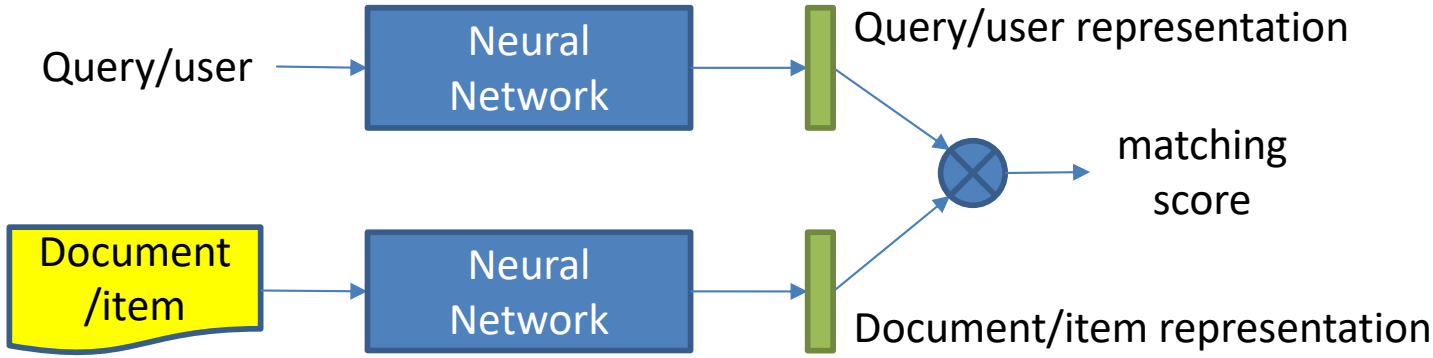
“Deep” Semantic Matching

- Representation
  - Word: one hot —> distributed
  - Sentence: bag-of-words —> distributed representation
  - Better representation ability, better generalization ability

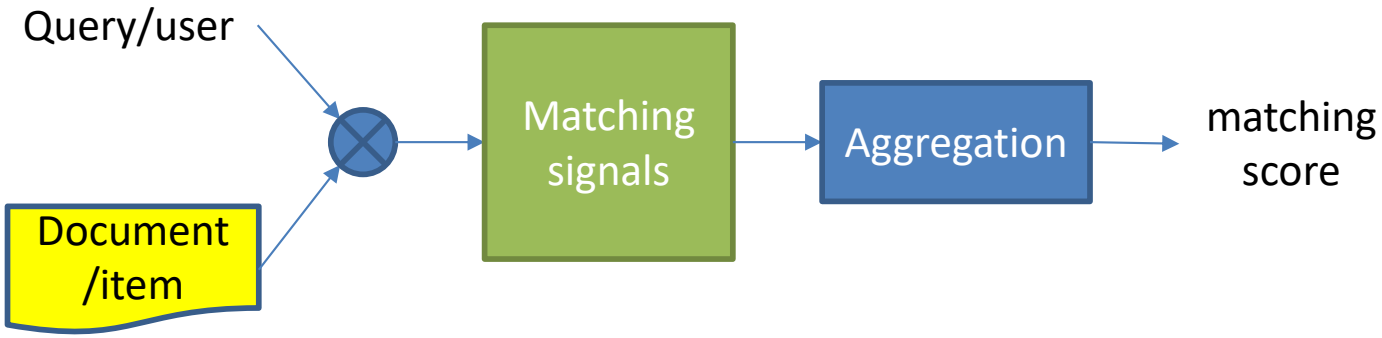
- Matching function
  - Inputs (features): handcrafted —> automatically learned
  - Function: simple functions (e.g., cosine, dot product) —> neural networks (e.g., MLP, neural tensor networks)
  - Involving richer matching signals
  - Considering soft matching patterns

Deep Learning Paradigms for Matching

- Methods of representation learning

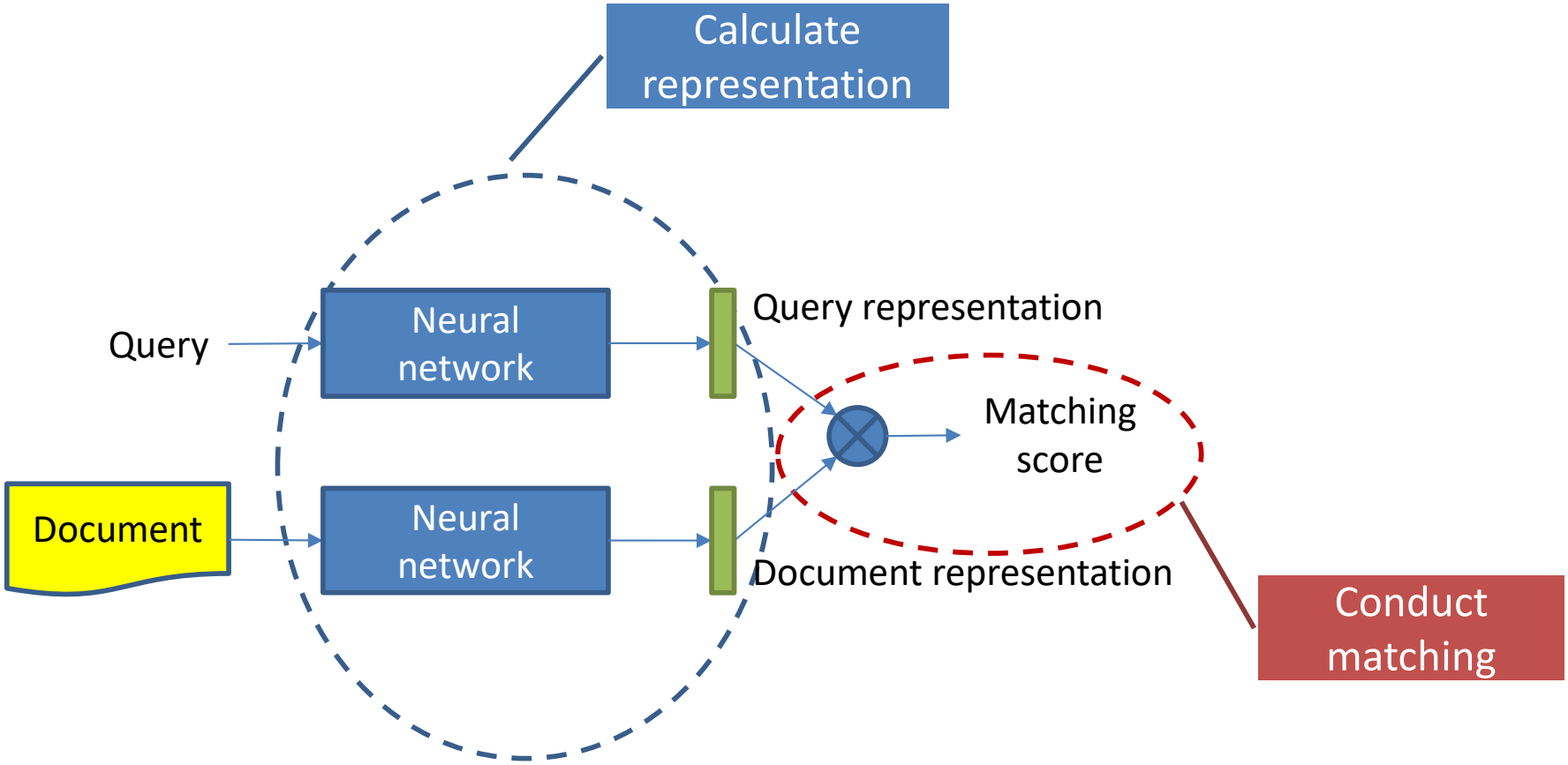


- Methods of matching function learning



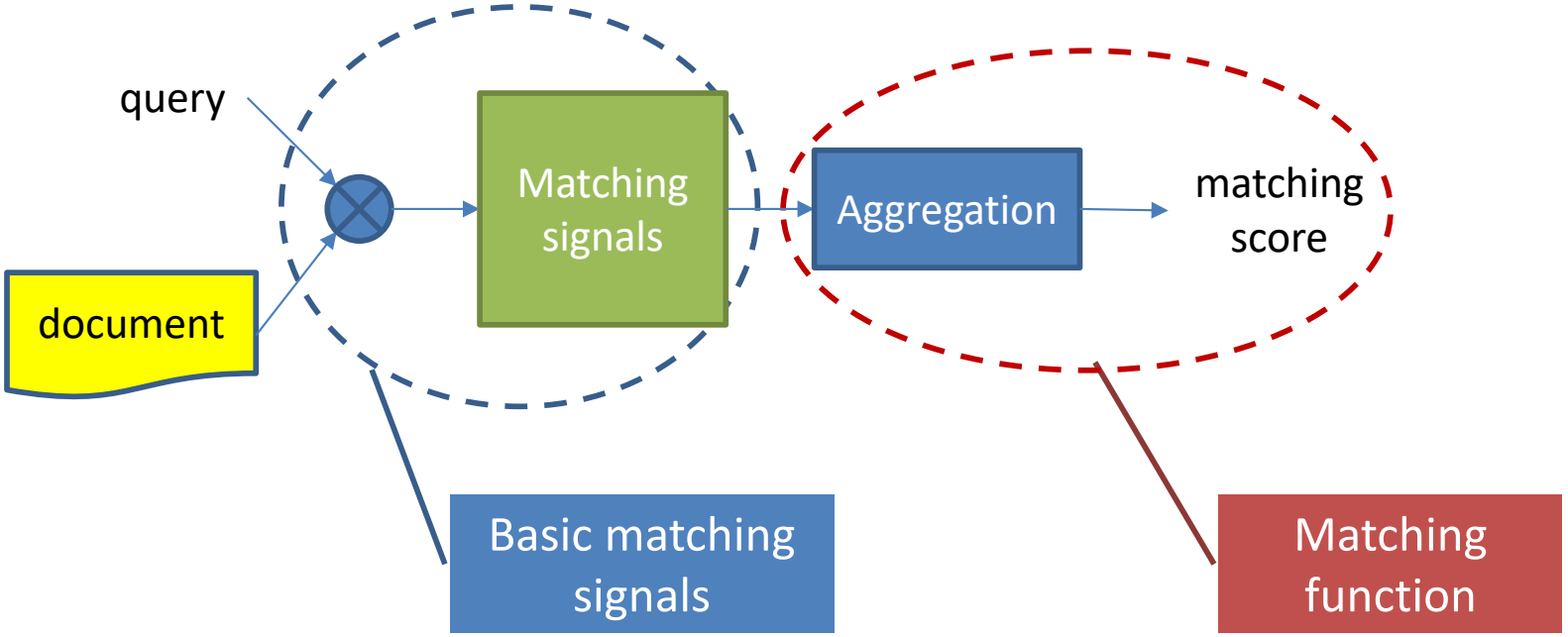
Methods of Representation Learning

- Step 1: calculate representation $\phi(x)$
- Step 2: conduct matching $F\big(\phi(x), \phi(y)\big)$



Methods of Matching Function Learning

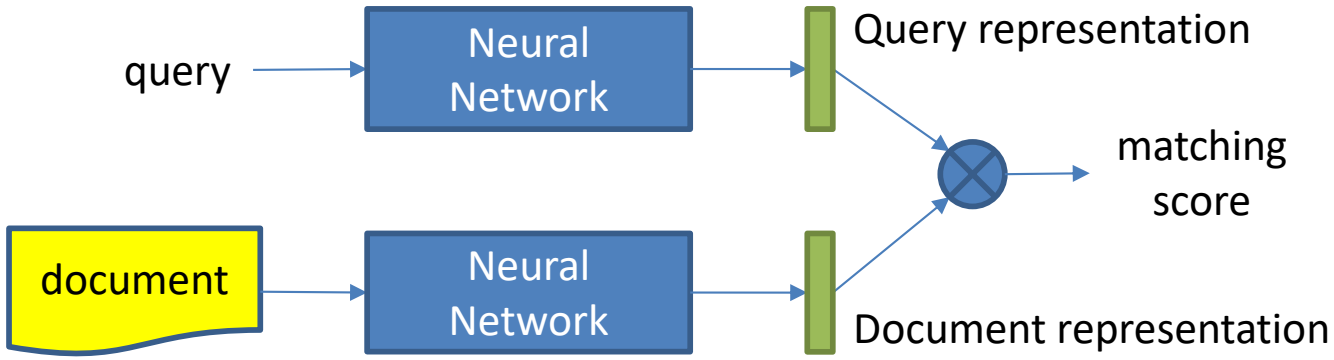
- Step 1: construct basic low-level matching signals
- Step 2: aggregate matching patterns



Outline of Tutorial

- Unified View of Matching in Search and Recommendation
- Part 1: Traditional Approaches to Matching
- Part 2: Deep Learning Approaches to Matching
  - Overview
  - Deep matching models for search
  - Deep matching models for recommendation
- Summary

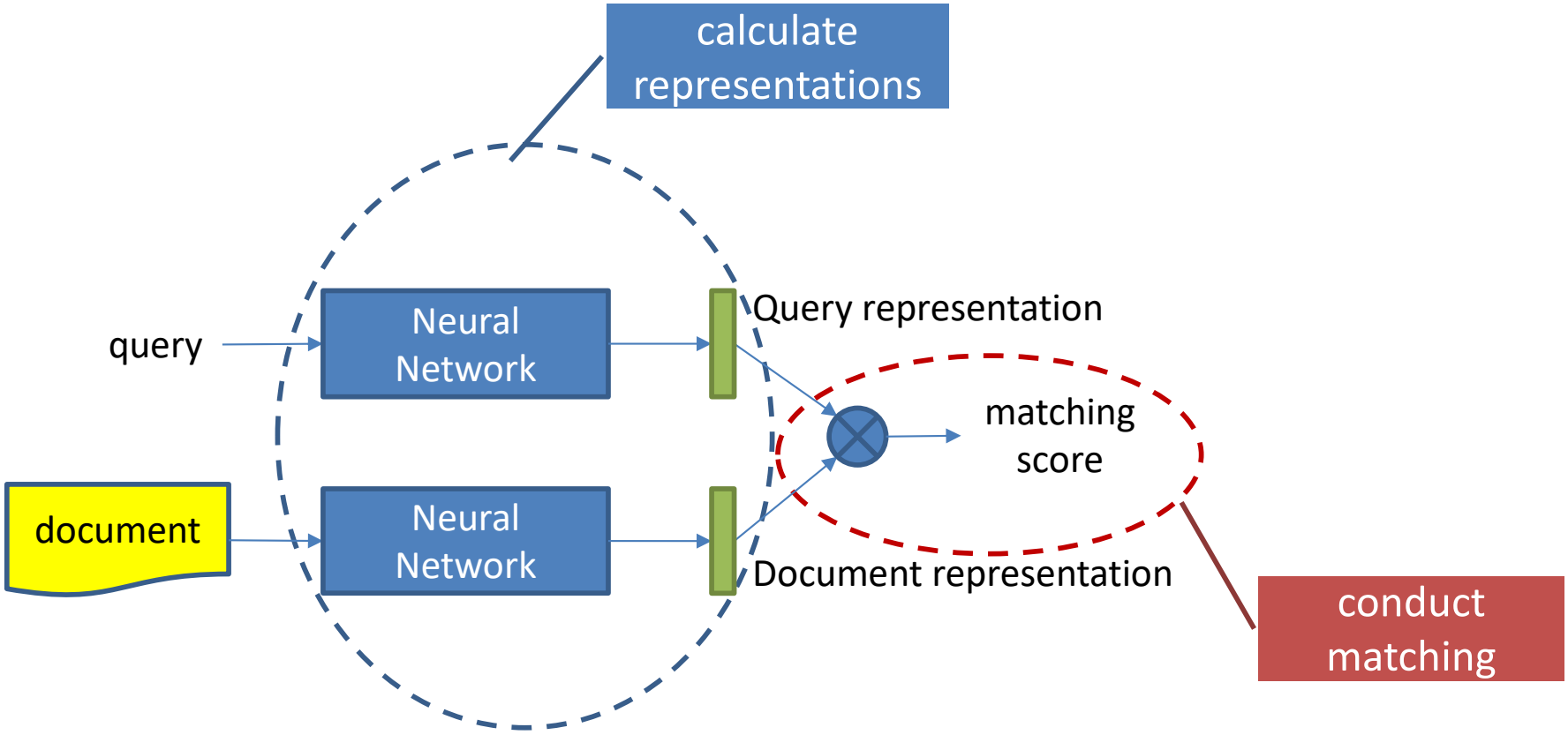
Slides: http://comp.nus.edu.sg/~xiangnan/sigir18-deep.pdf



METHODS OF REPRESENTATION LEARNING

Representation Learning for Query-Document Matching

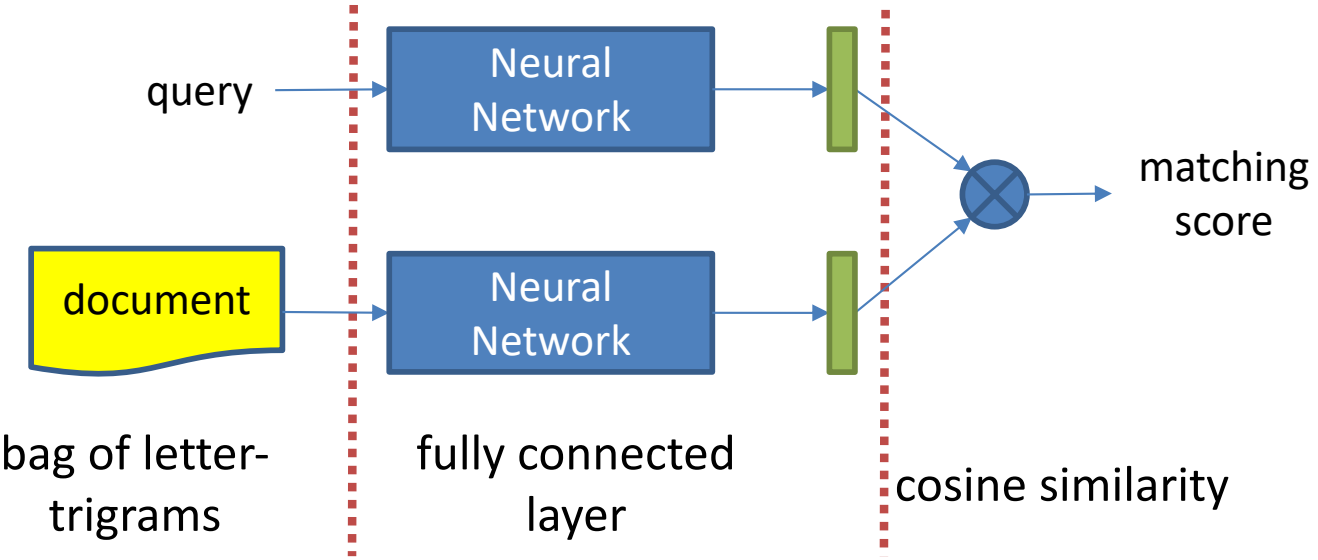
- Step 1: calculate query and document representation
- Step 2: conduct query-document matching



Typical Methods of Representation Learning for Matching

- Based on DNN
  - **DSSM**: Learning Deep Structured Semantic Models for Web Search using Click-through Data (Huang et al., CIKM ’13)
- Based on CNN
  - **CDSSM**: A latent semantic model with convolutional-pooling structure for information retrieval (Shen et al. CIKM ’14)
  - **ARC I**: Convolutional Neural Network Architectures for Matching Natural Language Sentences (Hu et al., NIPS ’14)
  - **CNTN**: Convolutional Neural Tensor Network Architecture for Community-Based Question Answering (Qiu and Huang, IJCAI ’15)
- Based on RNN
  - **LSTM-RNN**: Deep Sentence Embedding Using the Long Short Term Memory Network: Analysis and Application to Information Retrieval (Palangi et al., TASLP ’16)

Deep Structured Semantic Model (DSSM)



- Bag-of-words representation
  - "candy store": [0, 0, 1, 0, …, 1, 0, 0]
- Bag of letter-trigrams representation
  - "#candy# #store#" --> #ca can and ndy dy# #st sto tor ore re#
  - Representation: [0, 1, 0, 0, 1, 1, 0, …, 1]
- Advantages of using bag of letter-trigrams
  - Reduce vocabulary: #words 500K → # letter-trigram: 30K
  - Generalize to unseen words
  - Robust to misspelling, inflection etc.

DSSM Query/Doc Representation: DNN

- Model: DNN (auto-encoder) to capture the compositional sentence representations

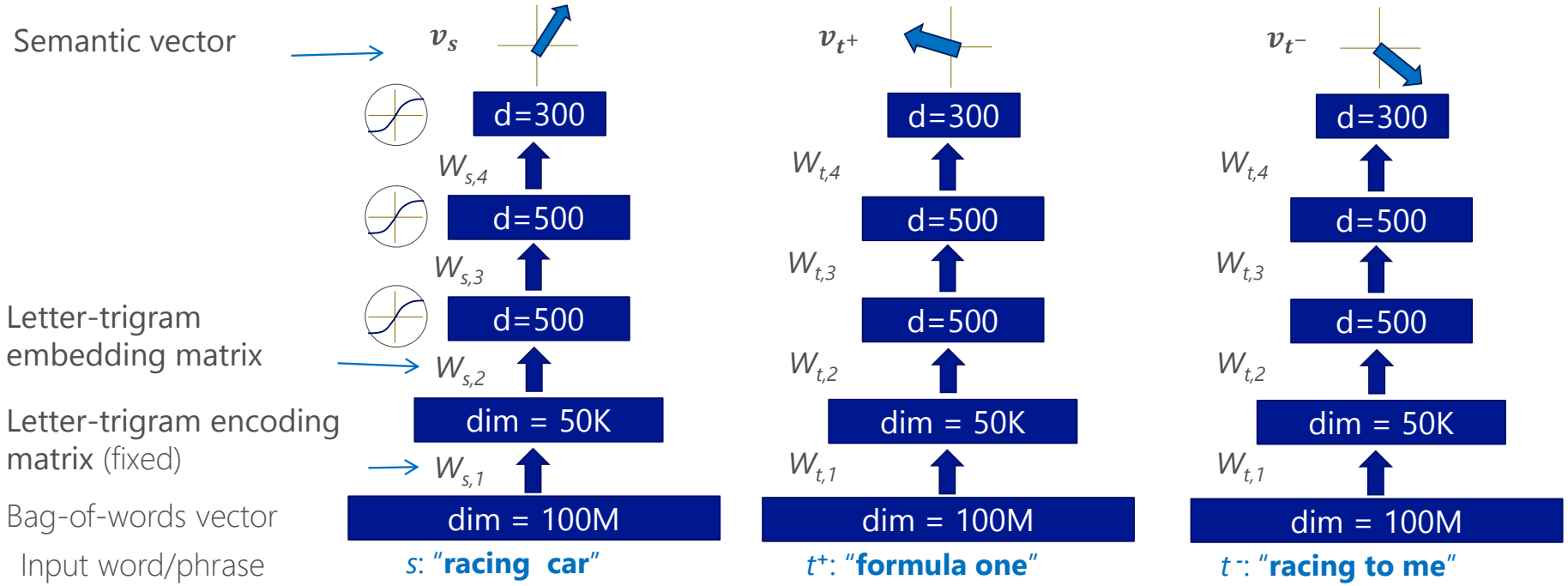


Figure courtesy of He et al., CIKM '14 tutorial

DSSM Matching Function

- Cosine similarity between semantic vectors

$$S = \frac{x^T \cdot y}{|x| \cdot |y|}$$

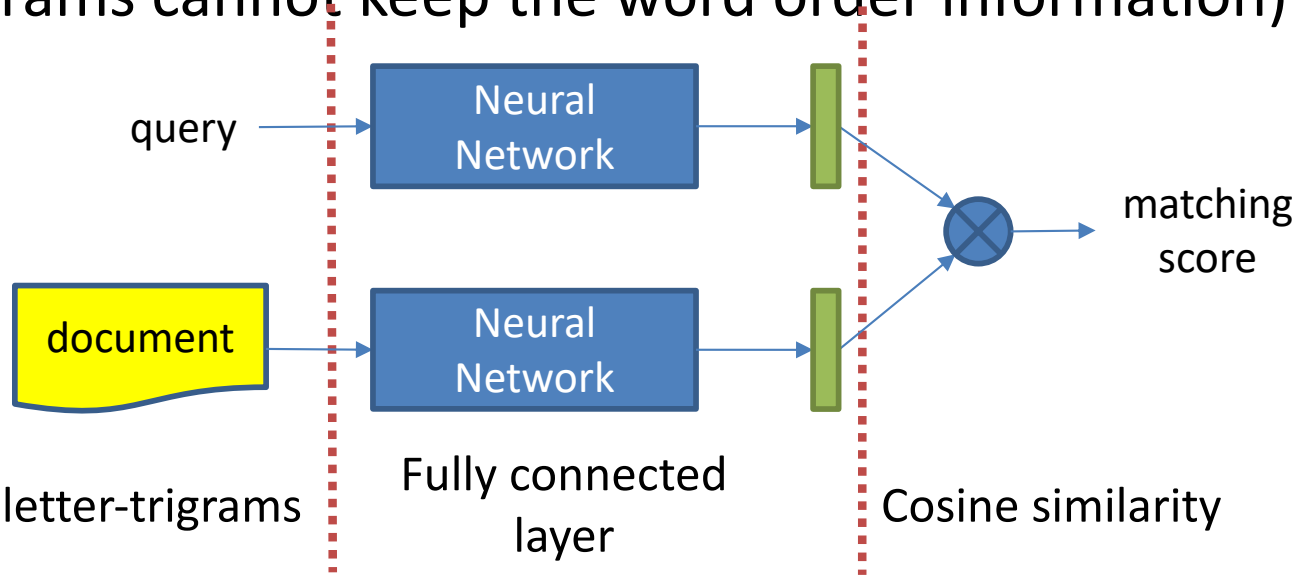
- Training
  - A query q and a list of docs $D = \{d^+, d_1^-, \cdots, d_k^-\}$
  - d^+ positive doc, $d_1^-, \cdots, d_k^-$ negative docs to query
  - Objective:

$$P(d^+|q) = \frac{\exp(\gamma \cos(q, d^+))}{\sum_{d \in D} \exp(\gamma \cos(q, d))}$$

DSSM: Brief Summary

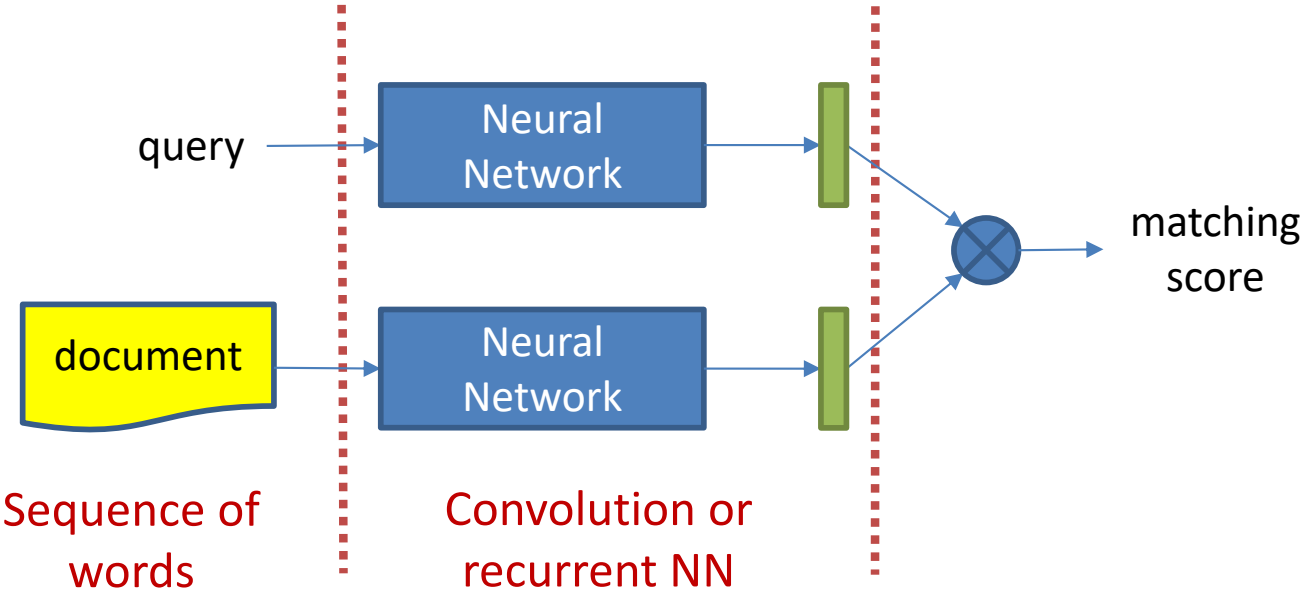
- **Inputs**: Bag of letter-trigrams as input for improving the scalability and generalizability
- **Representations**: mapping sentences to vectors with DNN: semantically similar sentences are close to each other
- **Matching**: cosine similarity as the matching function

- **Problem**: *the order information of words is missing* (bag of letter-trigrams cannot keep the word order information)



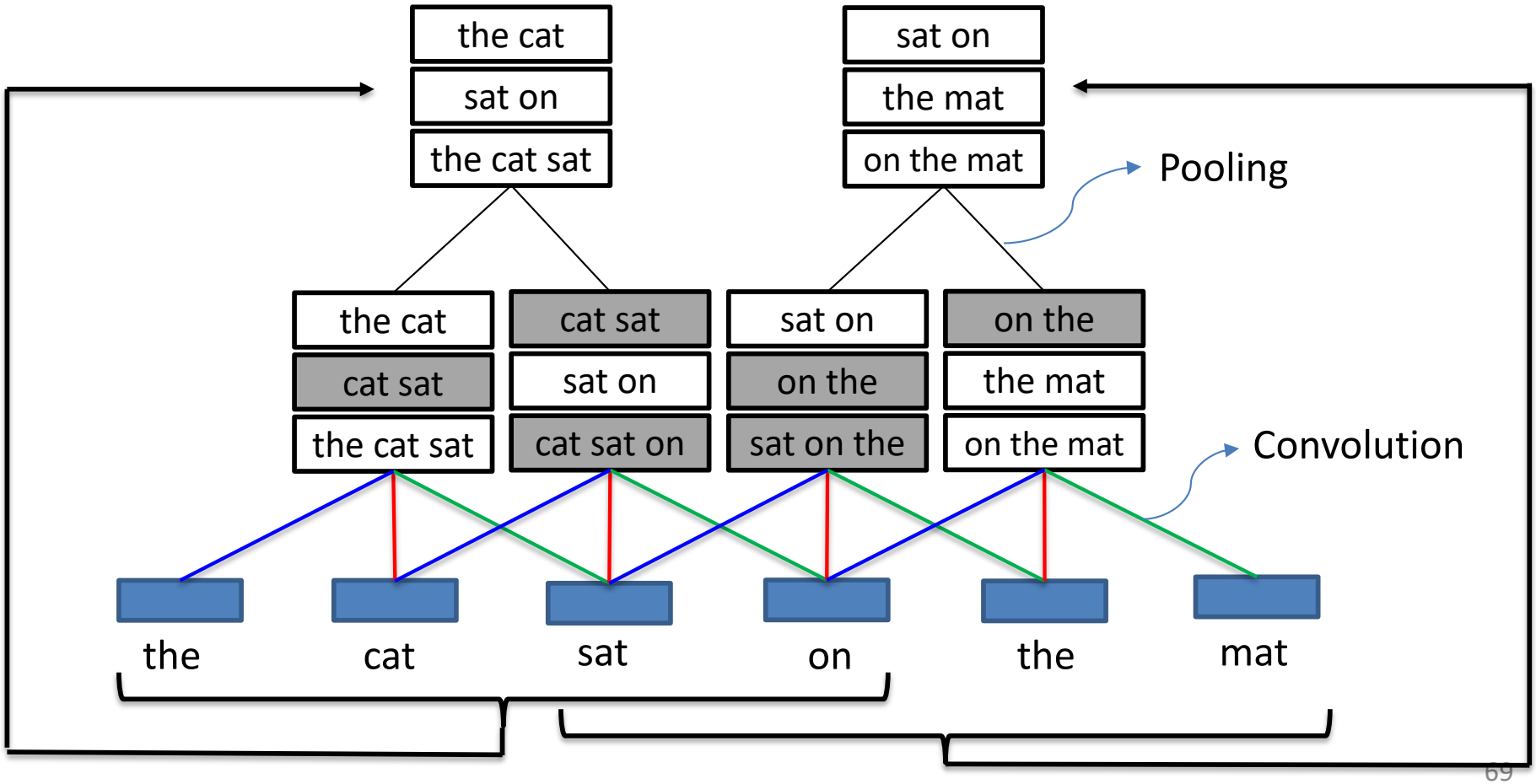
How to Capture Order Information?

- Input: **word sequence** instead of bag of letter-trigrams
- Model
  - **Convolution** based methods can keep locally order
  - **Recurrent** based methods can keep long dependence relations



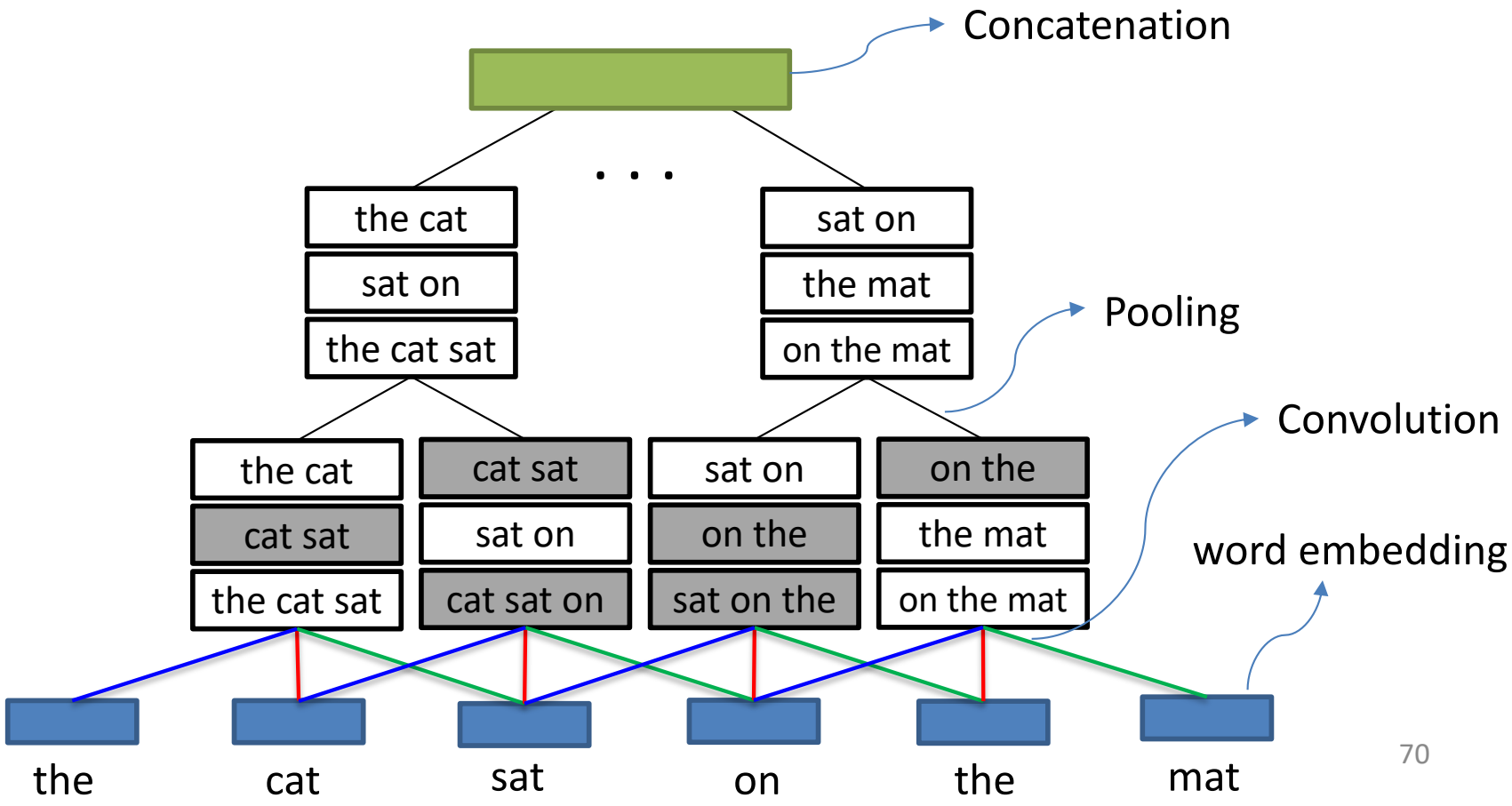
CNN Can Keep Order Information

1-D convolution and pooling operations can keep the local word order information



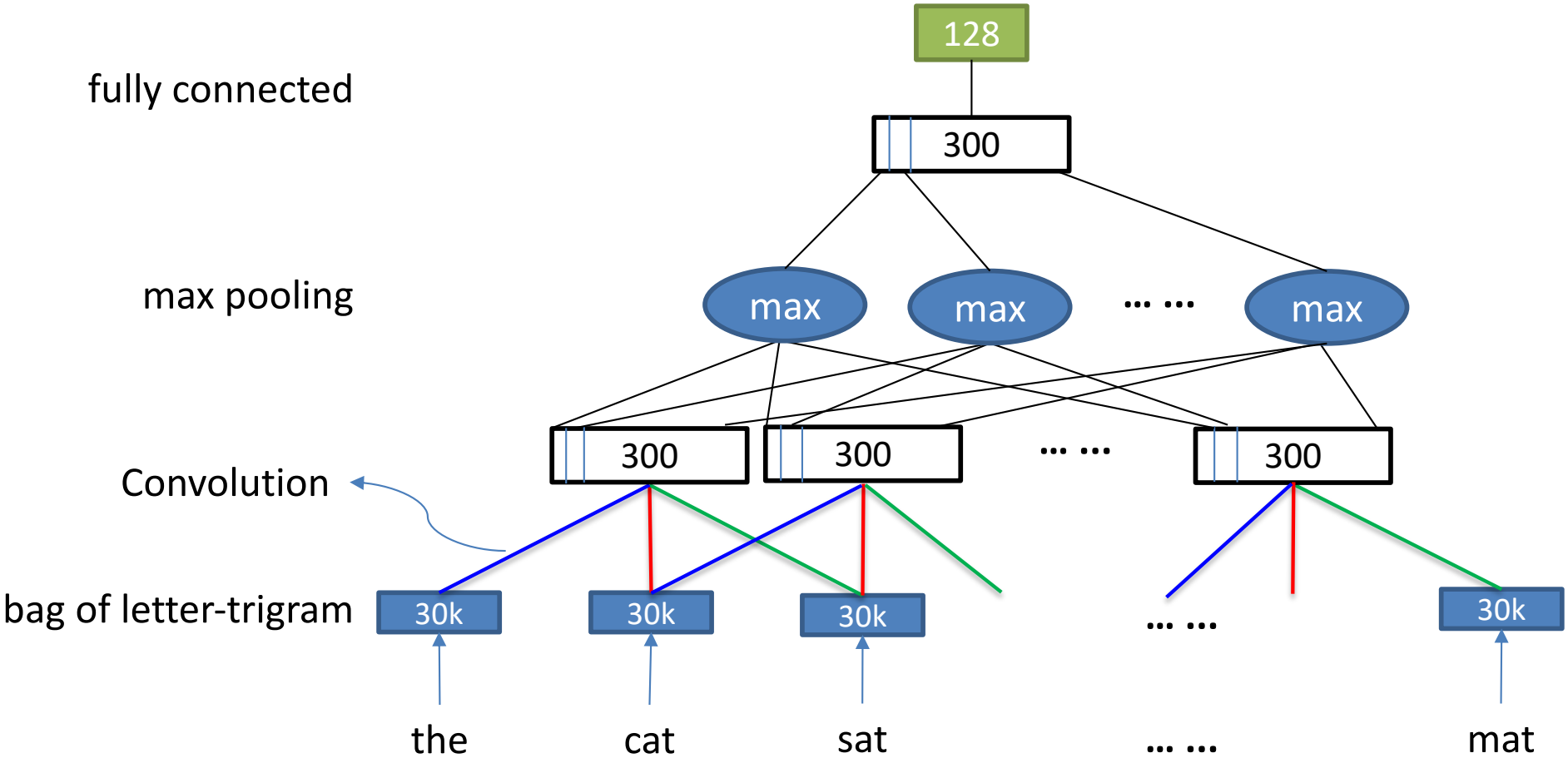
Using CNN: ARC-I (Hu et al., 2014) and CNTN (Qiu et al., 2015)

- Input: sequence of word embeddings trained on a large dataset
- Model: the convolutional operation in CNN compacts each sequence of k words

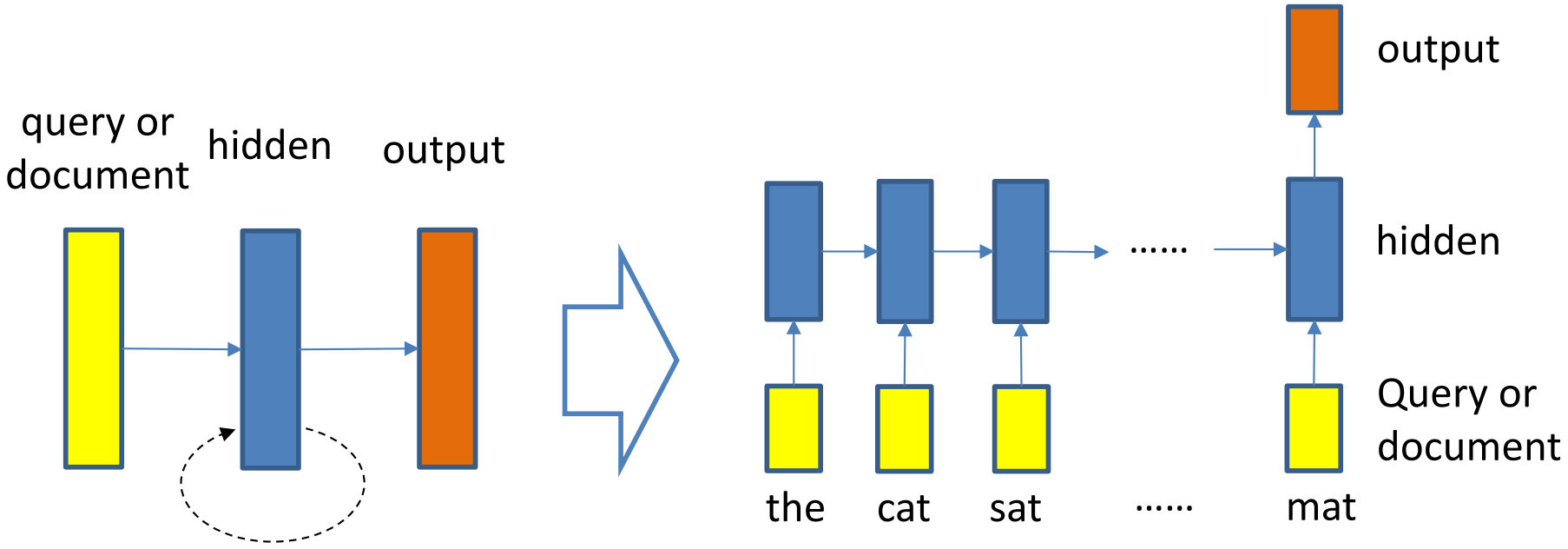


Using CNN: CDSSM (Shen et al., '14)

The convolutional operation in CNN compacts each sequence of k words



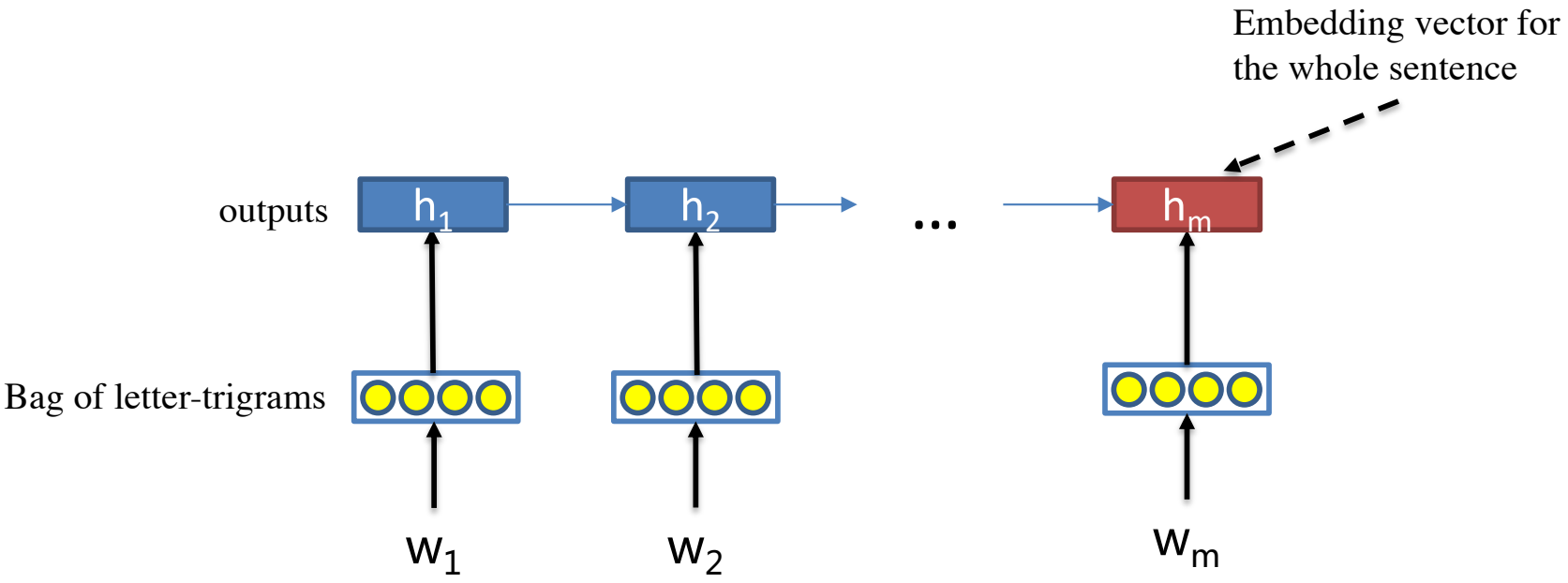
RNN can Keep the Order Information



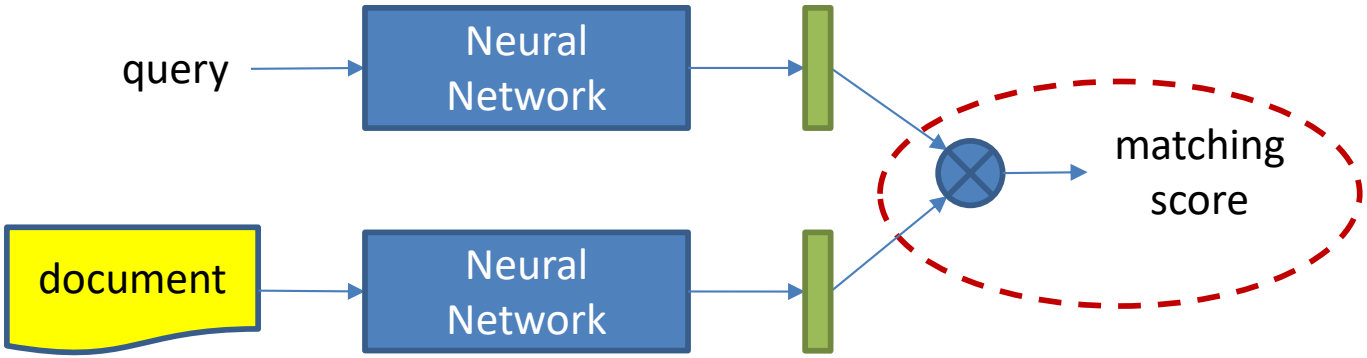
- Two popular variations: long-short term memory (LSTM) and gated recurrent unit (GRU)

Using RNN: LSTM-RNN (Palangi et al., ’16)

- Input: sequence letter trigrams
- Model: long-short term memory (LSTM)
  - The last output as the sentence representation



Matching Function



- **Heuristic**: cosine, dot product
- **Learning**: MLP, Neural tensor networks

Matching Functions (cont')

- Given representations of query and document : q and d
- Similarity between these two representations:
  - Cosine Similarity (DSSM, CDSSM, RNN-LSTM)

$$s = \frac{q^T \cdot d}{|q| \cdot |d|}$$

  - Dot Product

$$s = q^T \cdot d$$

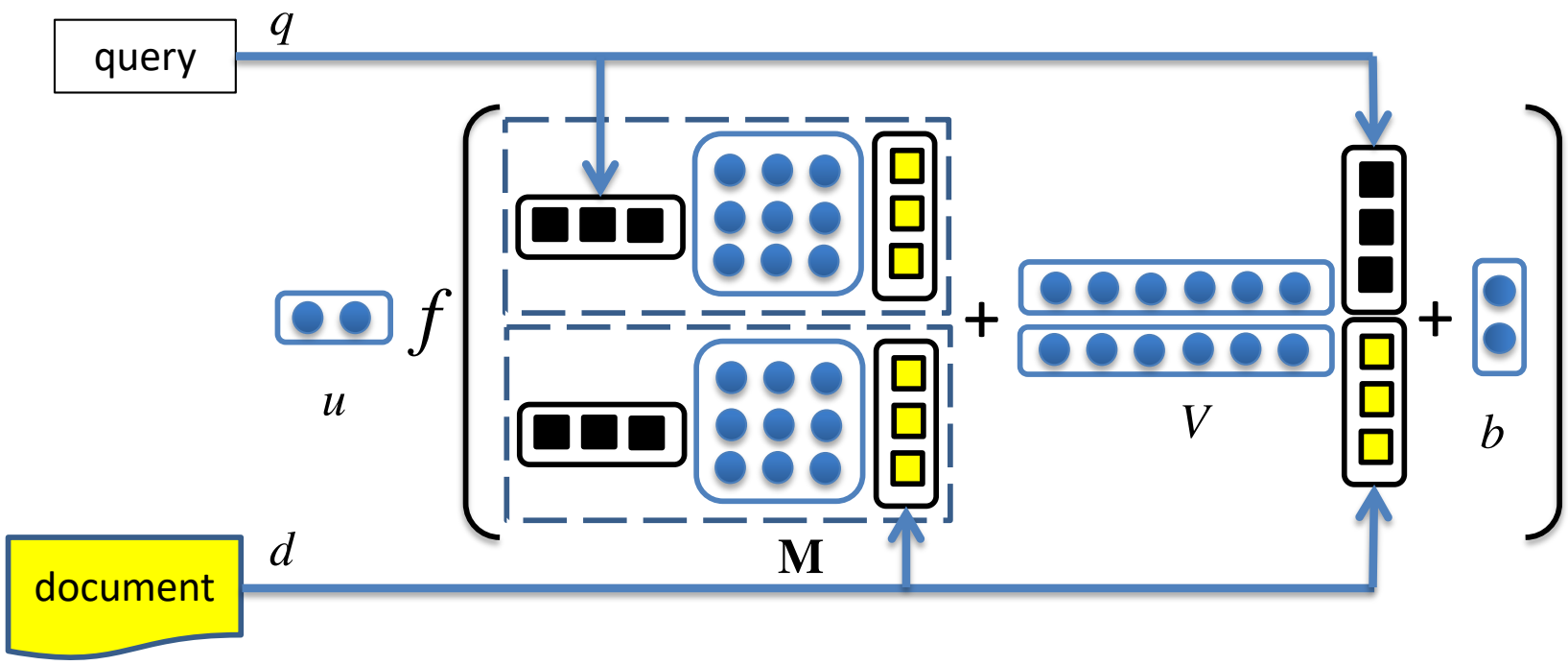
  - Multi-Layer Perception (ARC-I)

$$s = W_2 \cdot \sigma \left(W_1 \cdot \begin{bmatrix} q \\ d \end{bmatrix} + b_1 \right) + b_2$$

Matching Functions (cont')

- Neural Tensor Networks (CNTN) (Qiu et al., IJCAI '15)

$$s = u^T f\left(q^T \mathbf{M}^{[1:r]} d + V \begin{bmatrix} q \\ d \end{bmatrix} + b\right)$$



Experimental Results

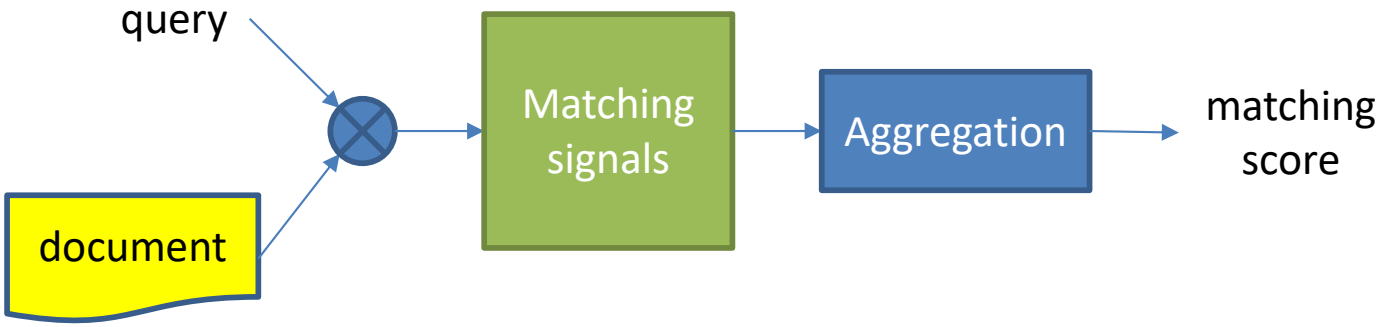
	Model	P@1	MRR
Traditional methods	BM25	0.579	0.726
Representation learning for matching	ARC-I	0.581	0.756
	CNTN	0.626	0.781
	LSTM-RNN	0.690	0.822

Based on Yahoo! Answers dataset (60,564 question-answer pairs)

- Representation learning methods outperformed baselines
  - Semantic representation is important
- LSTM-RNN performed better than ARC-I and CNTN
  - Modeling the order information does help

Short Summary

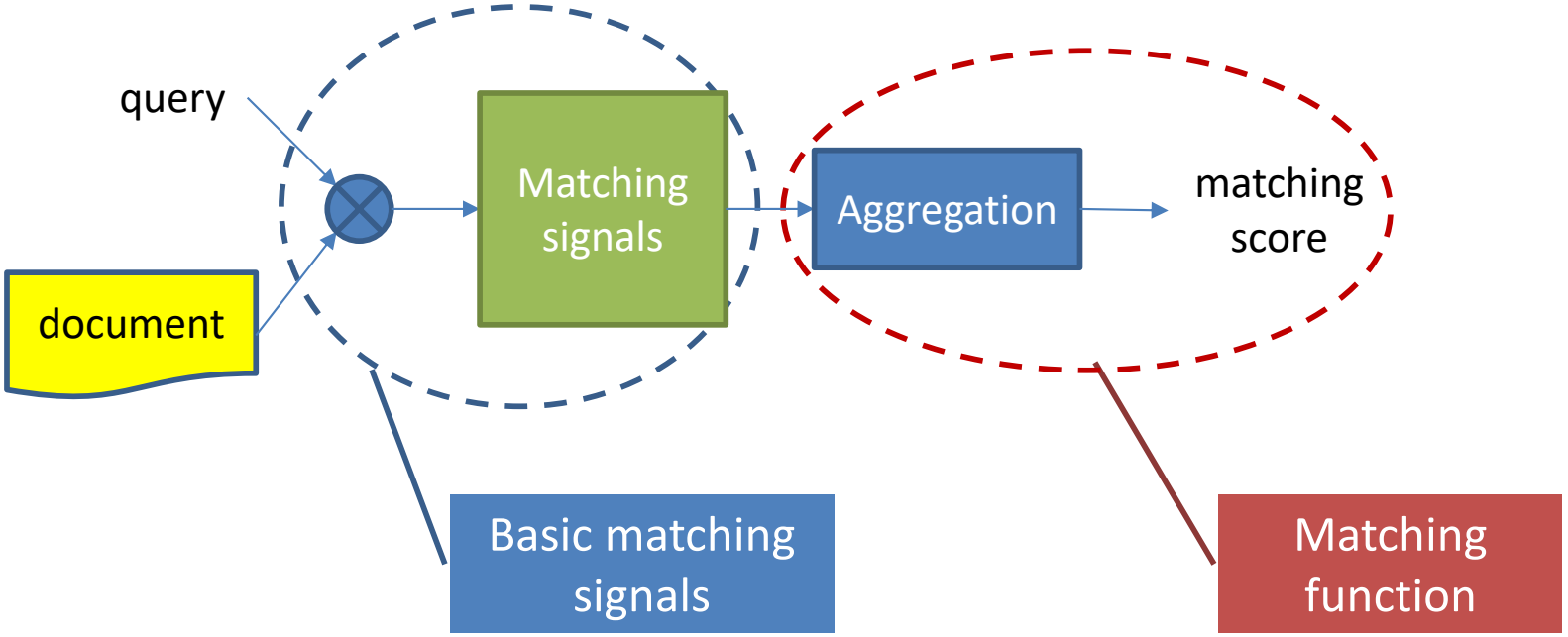
- Two steps
  - 1. Calculate representations for query and document
  - 2. Conduct matching
- Representations for query and document
  - Using DNN
  - Using CNN and RNN to capture order information
- Matching function
  - Dot product (cosine similarity)
  - Multi-layer Perceptron
  - Neural tensor networks



METHODS OF MATCHING FUNCTION LEARNING

Matching Function Learning

- Step 1: construct basic low-level matching signals
- Step 2: aggregate matching patterns



Typical Matching Function Learning Methods

- Matching with word-level similarity matrix
  - ARC II (Hu et al., NIPS ’14)
  - MatchPyramid (Pang et al., AAAI ’16)
  - Match-SRNN (Wan et al., IJCAI ’16)
- Matching with attention model
  - (Parikh et al., EMNLP ‘16)
- Combining matching function learning and representation learning
  - Duet (Mitra et al., WWW ’17)